UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

SISTEMA DE VIRTUALIZACIÓN Y GESTIÓN DE RECURSOS COMPARTIDOS EN UN GRUPO DE INVESTIGACIÓN UNIVERSITARIO

Manuel Jiménez Sánchez
Tutor: Juan Carlos San Miguel Avedillo
Ponente: José María Martínez Sánchez

Julio 2020

SISTEMA DE VIRTUALIZACIÓN Y GESTIÓN DE RECURSOS COMPARTIDOS EN UN GRUPO DE INVESTIGACIÓN UNIVERSITARIO

Manuel Jiménez Sánchez
Tutor: Juan Carlos San Miguel Avedillo
Ponente: José María Martínez Sánchez

Video Processing and Understanding Lab
Dpto. Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2020

Trabajo parcialmente financiado por el Ministerio de Economía y Competitividad del Gobierno de España bajo el proyecto TEC2017-88169-R MobiNetVideo (2018-2020).

Resumen

Este Trabajo de Fin de Grado tiene como propósito la creación de una infraestructura distribuida basada en virtualización para cálculo intensivo. Gracias a ello se obtiene un sistema robusto y eficiente, con una arquitectura modular que facilita la escalabilidad, la recuperación ante desastres y el respaldo de los datos. Este trabajo es un proceso íntegro, por lo que se describirán todos los pasos necesarios hasta llegar a la implementación final. Este proceso conlleva la evaluación de alternativas disponibles, el diseño lógico de la red y de los equipos, la instalación del sistema y los procedimientos necesarios para la puesta en marcha e implementación.

De manera paralela, se ha trabajado en una documentación técnica para que el personal pueda ajustar el sistema a sus necesidades, modificarlo y extenderlo. Del mismo modo se han elaborado múltiples guías que ayudan a los usuarios a utilizar los servicios que ofrece el sistema del laboratorio y que puedan ponerse en marcha rápidamente.

Los resultados de trabajo serán utilizados diariamente por los integrantes del grupo de investigación VPULab y es necesario que el sistema funcione correctamente para que esta infraestructura pueda facilitar el trabajo del personal investigador. Por esa razón, se realizará una evaluación de todos los componentes descritos para asegurar que el rendimiento es óptimo y que no se ha encontrado ninguna anomalía durante la implementación.

Palabras clave

Sistemas distribuidos, virtualización, contenedores, recursos compartidos, arquitectura de redes, Proxmox, KVM, LXC.

Abstract

This Bachelor Thesis aims at the creation of a complete infrastructure based on several virtualization and distributed system techniques. As a result, a robust and efficient system is achieved, including a modular architecture that simplifies scalability, disaster recovery and data backup. This work develops an integral solution, because of that, all necessary steps until the final implementation are described. Such process entails evaluation of the available alternatives, logic design of systems and network architecture, system installation and required procedures for the implementation of all components.

On the other side, a technical documentation has been elaborated for the staff to be capable to adjust the system to their needs by modifying and extending it. In the same way, multiple guides were written which help final users to utilize the services offered in the laboratory so they could quickly be able to get up and running.

This work's results will be used in a daily basis by the people in the research group so it is crucial that the system works as expected, simplifying the researcher's work. For that reason, all components will be evaluated to ensure optimal performance and the absence of anomalies during the process.

Keywords

Distributed systems, virtualization, containers, shared resources, network architecture, Proxmox, KVM, LXC.

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor Juan Carlos San Miguel por su gran ayuda y dedicación sin la cual este trabajo no habría podido realizarse y que me ha permitido desarrollar un proyecto que me ha aportado mucho más que lo que en un principio podría imaginar. A mis compañeros del VPULab que siempre han estado dispuestos a ayudar en lo que necesitase. También a todas aquellas personas que han contribuido a que esta situación sanitaria excepcional sea mucho más llevadera y que han permitido que el mundo siga girando. Por último, quiero mostrar mi agradecimiento a mi familia y amigos, que durante todos estos años me han ofrecido su apoyo y que gracias a ello, me han ayudado a seguir adelante en aquellos momentos de agotamiento.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

Las tecnologías de la información han sufrido cambios significativos en los últimos años. Son ya muy pocos los dispositivos que basan su funcionalidad en el almacenamiento local de la información. Los sistemas ya no dependen únicamente de un único dispositivo. La mayoría de servicios de *streaming*, los *smartphones* o los *wearables* se conectan a una infraestructura remota que permite la catalogación, el respaldo y la interacción de manera transparente para el usuario.

En el campo de la investigación, estas infraestructuras son cruciales, ya que es necesaria una gran capacidad de computación para llevar a cabo tareas de cálculo de la forma más eficiente posible. Por este motivo, es necesario un sistema que esté siempre disponible y que pueda tolerar altas cargas de trabajo de manera simultánea por el personal investigador.

Además de ello, es imprescindible una red bien planificada y construida, ya no solo para garantizar la concurrencia entre usuarios, también para que la infraestructura pueda ser reutilizable y extensible a medida que las demandas de la capacidad de procesamiento aumenten.

El estado anterior de la infraestructura del laboratorio, era un sistema centralizado en el que todos los servicios del laboratorio se suministraban a través de una máquina, al no estar dichos servicios modularizados, la realización de respaldos y la recuperación ante fallos era más difícil de garantizar.

Es por ello que el enfoque ha sido trabajar en dicha modularización para asegurar la disponibilidad del laboratorio en todo momento. Este trabajo está situado dentro del marco del proyecto de investigación *TEC2017-88169-R MobiNetVideo*. Por ese motivo, ciertas partes del proyecto ya existentes como el análisis de requisitos o el diseño de la arquitectura lógica han servido como punto de partida para el posterior desarrollo de este trabajo.

1.2. Objetivos

Como ya se ha comentado antes, se ha partido de una planificación de requisitos y de arquitectura lógica, aportados por el proyecto de investigación. Conforme a ese punto de partida se ha desarrollado una infraestructura de gestión de información y de capacidad de procesamiento. El sistema de modularización utilizado es la virtualización, previamente a su implementación se realizará un análisis previo de las posibles

soluciones de virtualización.

De manera paralela, se realizarán pruebas de rendimiento y disponibilidad a medida que se vayan añadiendo características al sistema, realizando análisis de redes y sometiendo a cargas de trabajo exhaustivas a los sistemas para verificar que su comportamiento es el correcto. De la misma forma, se trabajará en la elaboración de una documentación a disposición del laboratorio para que se pueda retomar y extender el trabajo realizado. Por último, se trabajará en alternativas para que los usuarios puedan gestionar su información interna de la forma más sencilla posible, minimizando los pasos requeridos para ello.

Son muchos los aspectos a tener en cuenta en este trabajo y es necesario que todos ellos funcionen de manera coordinada para asegurar que la infraestructura es sólida, extensible y que en general, se pueda utilizar en el día a día de las necesidades del grupo de investigación.

1.3. Organización de la memoria

Esta memoria consta de los siguientes capítulos:

- Capítulo 1 Introducción y objetivos para la realización del trabajo.
- Capítulo 2 Estado del arte. Presentación y comparativa de las alternativas disponibles para la modularización.
- Capítulo 3 Planteamiento del problema y descripción de los métodos para llevar a cabo el diseño de la infraestructura.
- Capítulo 4 Descripción de los procedimientos necesarios para implementar todos los aspectos planteados en la fase de diseño.
- Capítulo 5 Evaluación y análisis de las principales características de la infraestructura.
- Capítulo 6 Conclusiones finales conforme a los resultados obtenidos y posibles líneas de trabajo futuro.
- Apéndice A Detalle de las plataformas de virtualización analizadas.
- Apéndice B Casos de prueba detallados de los casos de uso más importantes del sistema.

Capítulo 2

Estado del arte

En este capítulo se analizarán dos tipos de tecnologías. En primer lugar se hará una comparativa de las principales tecnologías de virtualización y de contenedores. Muchas de estas opciones se utilizarán como base en el siguiente tipo de alternativas analizadas, correspondientes a las plataformas de virtualización. El objetivo de estas plataformas es ofrecer una solución íntegra que incluya más características que la propia emulación de sistemas, facilitando así su mantenimiento.

2.1. Tecnologías de virtualización

Para llevar a cabo abstracciones del software que sean aisladas y fácilmente transferibles a otras máquinas sin alterar su funcionamiento, se pueden contemplar dos tipos de tecnologías. En primer lugar, la virtualización es una tecnología que consiste en la emulación de recursos para la ejecución de una máquina virtual bajo un sistema operativo ya existente. Una máquina virtual puede emular tanto arquitecturas como sistemas operativos completamente diferentes al del sistema anfitrión. Algunos ejemplos de ello son KVM (*www.linux-kvm.org*), Xen (*www.xenproject.org*) o QEMU (*www.qemu.org*). Por otro lado, un contenedor se basa en un entorno de ejecución implementado como un proceso aislado del resto en la máquina host. Al contrario que una máquina virtual, un contenedor es más ligero, ya que puede compartir sus recursos con la máquina principal a medida que se necesiten. En la actualidad existen tecnologías de contenedores como Docker (*www.docker.org*), Kubernetes (*kubernetes.io*), OpenVZ (*www.openvz.org*) o LXC (*linuxcontainers.org*). Cabe destacar que se prefieren soluciones completas y que terceras personas puedan configurar aspectos básicos de los sistemas a través de una interfaz visual.

En la Tabla 2.1, se detallan las características de las tres tecnologías de virtualización analizadas. Se han elegido tecnologías libres debido a que las tecnologías comerciales suelen incluirse en conjunto con el producto ofrecido y la documentación de las mismas es más escasa. Por un lado KVM está orientado a la virtualización completa, es decir, que busca que la virtualización esté lo más cerca posible del núcleo y del hardware de la máquina huésped para mejorar su rendimiento. QEMU, basado en el código de KVM, se centra más en la emulación del mayor número de arquitecturas posible. Como alternativa a KVM existe Xen, un hipervisor que se ejecuta sobre un sistema operativo ya existente. Esto provoca que su rendimiento sea un poco menor que KVM en operaciones que requieran interacción con el hardware de la máquina.

En la Tabla 2.2 se analizan distintas tecnologías de contenedores. La más extendida

	KVM	Xen	QEMU
Paravirtualización (SW)	•	•	•
Virtualización (HW)	•	•	
Arquitecturas emuladas	x86, x86_64, IA_64, PPC	x86, x86_64, ARM	x86, MIPS, x86_64, ARM, PPC, SPARC
SS.OO. emulados	Windows, Linux, BSD, UNIX	Windows, Linux, BSD, UNIX	Windows, Linux, BSD, UNIX
Discos	raw, qcow2, vmdk	raw, vhd, vmdk	raw, qcow2, vmdk
Máx. núcleos	384	32	Variable según plataforma
Máx. RAM	4 TB	1.5 TB	Variable según plataforma
VT-x/AMD-V	Obligatorio	Opcional	Opcional

Tabla 2.1: Tabla comparativa entre las principales características de las distintas tecnologías de virtualización.

a día de hoy es Docker, un sistema multiplataforma que cuenta con un gran soporte de la comunidad en cuanto a utilización y documentación. Utiliza un sistema de capas. Cada capa representa un cambio importante en el sistema de archivos del contenedor y pueden ser compartidas en otras imágenes. Docker puede ser utilizado con el servicio Docker Hub para la descarga de imágenes y junto con Docker Swarm, se pueden llevar a cabo técnicas de clusterizado. Kubernetes es un sistema de orquestación de contenedores que permite escalar la capacidad de los mismos conforme a la demanda requerida, por tanto, se pueden utilizar dentro de un clúster y es compatible con las imágenes de Docker. Por otro lado, OpenVZ es una tecnología de contenedores para Linux implementado como un proceso más del sistema, pero que se encuentra aislado del mismo. De la misma forma, LXC está implementado de manera parecida a OpenVZ, pero cuenta con una gran ventaja: su inclusión por defecto en cualquier sistema que use el núcleo Linux.

	Docker CE	LXC	OpenVZ	Kubernetes
Incluido en kernel Linux		•		
Migraciones en vivo			•	
Clustering	Docker Swarm			•
Repositorios de imágenes	Docker Hub	linuxcontainers.org	openvz.org	Compatibles con docker
Licencia	Apache	GPLv2	GPLv2	Apache

Tabla 2.2: Tabla comparativa entre las principales características de las distintas tecnologías de contenedores.

2.2. Plataformas de virtualización

En la actualidad existen multitud de plataformas software que proporcionan soluciones completas de entornos de virtualización. Todas ellas cuentan con determinadas características que son cruciales dependiendo del entorno que se administre. Algunos trabajos han descrito distintos paradigmas de virtualización para laboratorios de investigación [1][2]. En este apartado se analizarán algunas de las plataformas más importantes, contemplando tanto opciones libres como opciones propietarias y si las plataformas son *bare-metal* [1].

En el Anexo A se analizan con más detalle el conjunto de las alternativas analizadas, que comprenden tanto plataformas comerciales (VMWare, Hyper-V o vmmanager)

[1]Una plataforma *bare-metal* hace un uso completo de todos los recursos disponibles de la máquina en la que se instala, siendo dicha plataforma la encargada de gestionar esos recursos hacia los distintos hipervisores o usuarios.

como alternativas de código abierto (Virtualbox, KVM, Ovirt, Archipel, Proxmox o Citrix). En el siguiente apartado se detallarán las conclusiones de este análisis.

2.3. Comparativa

	VMware	Hyper-V	vmmanager	VirtualBox	KVM	Ovirt	Archipel	Proxmox	Citrix
Código abierto				•	•	•	•	•	
Arquitectura x86_64	•	•	•	•	•	•	•	•	•
Soporte NFS nativo	•		•			•		•	•
Backups Periódicos	•	•	•			•		•	•
Migraciones en vivo	•	•	•	•		•	•	•	•
Contenedores	•		•				•	•	•
Administración web	•		•			•	•	•	•
Bare-metal	•		•			•	•	•	•
Precio	967,50€	972€	4€/pc/mes	Gratis	Gratis	Gratis	Gratis	Gratis	693,32€
Última actualización	Dic 2019	2019	2020	Feb 2020	Mayo 2020	Marzo 2020	Mar 2015	Mayo 2020	Abril 2020

Tabla 2.3: Tabla comparativa entre las principales características de las distintas plataformas de virtualización.

En un primer lugar se descartan VirtualBox y KVM que no son bare-metal, es decir, necesitan ejecutarse sobre un sistema operativo ya instalado. Esto provoca que no se aprovechen totalmente los recursos hacia la virtualización y que el rendimiento no sea tan bueno como en el resto de alternativas.

Las alternativas de pago suelen incluir un soporte técnico adicional en el precio que para este tipo de infraestructura no necesitaremos. Además, en muchos casos las licencias de uso son por zócalo de CPU, por lo que los costes en un futuro a la hora de escalar la red podrían elevarse demasiado. Por ello consideraremos las opciones de VMWare, Hyper-V, Vmmanager y Citrix como descartadas.

Restan tres alternativas que son las que más se ajustan a nuestros requisitos: Archipel, Ovirt y Proxmox. La última actualización de Archipel data de Marzo de 2015 por lo que existe la posibilidad de que el proyecto haya sido abandonado. En el caso de Ovirt, que sigue siendo mantenida actualmente, se encuentran algunos inconvenientes: Es un sistema focalizado en almacenamiento distribuido por lo que integrar unidades locales requiere configuración adicional. Además, la gestión de contenedores LXC no está implementada en la actualidad.

En los últimos años, Proxmox está adquiriendo relevancia en todo tipo de entornos. Soporta tanto unidades remotas como unidades locales. Su interfaz permite gestionar clústers, unidades de disco y permite asignar recursos a usuarios individuales, aislándolos del resto de la máquina. Cuenta con soporte de contenedores LXC y el uso de plantillas. La virtualización se proporciona a nivel de kernel usando KVM. Proxmox integra tecnologías open-source que ya se usan en entornos de producción y su documentación es bastante extensa[2]. Además Proxmox soporta de serie la implementación en Linux del sistema de ficheros ZFS, un sistema de ficheros robusto, completo y con soporte RAID que, usualmente debido a licencias incompatibles, no suele incluirse en las principales distribuciones Linux. Una de las más grandes fortalezas de Proxmox es su potente interfaz de web de administración que permite que usuarios noveles con el sistema puedan realizar operaciones básicas de migración y respaldo sin tener que

[2]Promxox VE documentation *https://pve.proxmox.com/wiki/Main_Page* (Visitado: 2 Abril 2020)

memorizar ningún comando específico. Por todo ello, se ha elegido Proxmox [3] como el sistema que estará instalado en los servidores de la red.

Capítulo 3

Diseño

3.1. Introducción

Después de evaluar el estado del arte se ha realizado un análisis de requisitos para conocer cuál es el alcance del proyecto y las necesidades específicas de los usuarios. Una vez realizado dicho análisis, se procederá a la implementación final de dichos requerimientos.

3.2. Análisis de Requisitos

RF1. El sistema proporcionará múltiples servicios para la gestión de la red.

- **RF1.1.** La red dispondrá de una puerta de enlace que garantice la interconexión de las distintas subredes existentes.
- **RF1.2.** Se utilizará una herramienta de código abierto para la gestión de usuarios.
- **RF1.3.** Se otorgará acceso remoto a los sistemas de manera segura e individualizada para cada usuario.
- **RF1.4.** Se dispondrá de un servicio que mantenga sincronizados todos los relojes de los equipos de la red de gestión.
- **RF1.5.** El sistema deberá integrar un servicio que pueda resolver los nombres de las máquinas de la red.
- **RF1.6.** Los usuarios podrán acceder a Internet desde sus máquinas virtuales mediante un Proxy.

RF2. Se contará con una documentación para asegurar el mantenimiento del sistema a largo plazo.

- **RF2.1.** Se explicarán los procedimientos en una *wiki* privada a disposición de los administradores de la red.
- **RF2.2.** Se utilizará una *wiki* pública donde se explicarán las instrucciones de uso de los servicios por parte de los usuarios finales.
  - **RF2.2.1.** Se documentará el acceso remoto a la red.
  - **RF2.2.2.** Se explicará cómo acceder a los recursos compartidos del grupo de investigación.

RF3. Existirán recursos compartidos entre los distintos usuarios del grupo de investigación.

RF3.1. Los distintos recursos compartidos estarán disponibles desde cualquier equipo situado dentro de la red de la Universidad Autónoma de Madrid.

RF3.2. Los recursos compartidos estarán disponibles desde cualquier ordenador de la red interna del grupo de investigación.

RF4. Se monitorizarán los servicios implementados con una herramienta adecuada para ello.

RF4.1. En caso de que algún servicio falle, se informará de ello de manera automática.

RF4.2. Se aislarán los servicios para que en caso de fallo grave el impacto sea mínimo.

RF5. Se deberán realizar copias de seguridad periódicamente.

RF5.1. Se dispondrán de copias en las propias máquinas cada tres días.

RF5.2. Se realizarán copias en un servidor remoto una vez por semana.

RF6. Los recursos disponibles serán aprovechados al máximo y al mismo tiempo se distribuirán entre varios sistemas para distribuir su carga.

3.3. Arquitectura Lógica

Como se puede ver en la Figura 3.1, el sistema contará con distintas subredes dependiendo de la finalidad de los dispositivos que se encuentran en su interior.

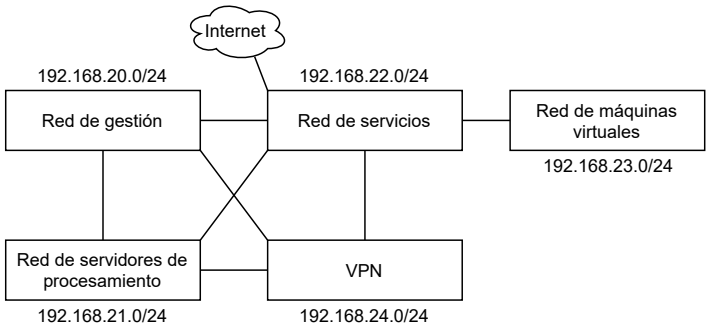


Figura 3.1: Diagrama básico de las principales subredes y su interconexionado.

- Se establecerá una **subred de gestión**, orientada a la comunicación interna del clúster de los **servidores físicos en los que se alojarán todos los servicios**. Dispondrá de un switch exclusivo para asegurar siempre el buen estado del clúster de los tres servidores. a esta subred se le asignará el número 20.
- Todos los servidores físicos, independientemente de su finalidad, se situarán en una **subred de servidores de procesamiento**, que será identificada como la subred 21.

- Una **subred de servicios** alojará todos los servicios necesarios por los usuarios del laboratorio. Se situará en la subred 22.
- La **subred de máquinas virtuales** alojará las máquinas virtuales y contenedores utilizados por los usuarios finales del laboratorio se situarán en la subred 23.
- Los usuarios que se conecten a través de la VPN tendrán una IP asignada en la **subred 24**.

3.3.1. Subred de gestión

Para alojar todos los servicios necesarios de la red, se utilizarán tres servidores: dos de ellos exclusivamente para servicios de gestión y uno adicional que será utilizado también como almacenamiento compartido. Estos tres servidores se hallarán organizados en un clúster para poder transferir servicios en ejecución de manera transparente. Para ello se utilizará un switch orientado exclusivamente al tráfico generado por dicho clúster tal y como se muestra en la Figura 3.2.

Los servidores *vpunet-mgmt01-002* y *vpunet-mgmt02-003* serán los únicos que dispondrán de conexión a Internet, debido a ello habrá que implementar distintos servicios para que ciertas funcionalidades en los otros servidores puedan funcionar con normalidad.

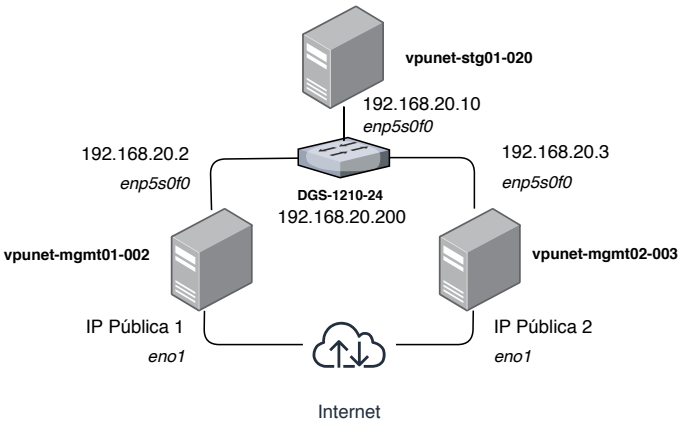


Figura 3.2: Diagrama de los componentes de la subred de gestión y de los distintos servidores que compondrán el clúster de gestión *vpunet-mgmt*.

3.3.2. Red de servidores de procesamiento

Además de la subred para el clúster de gestión, que se llamará *vpunet-mgmt*, estos tres servidores mencionados previamente se encontrarán en una subred de servidores de procesamiento, junto a otros servidores destinados a máquinas de trabajo, que serán parte de otro clúster llamado *vpunet-work*. Para garantizar una cierta escalabilidad y un orden en un futuro, dentro de la subred se hará una distinción para cada tipo de servidor y su cometido en la red. El servidor de discos, *vpunet-stg01-010* estará conectado al switch de servidores de procesamiento con tres enlaces mediante *Link Aggregation* (LACP 802.3ad) para conseguir cierta tolerancia a los fallos y disminuir la saturación de las interfaces en casos de uso intensivo.

- Los servidores orientados al **suministro de servicios** tendrán direcciones IP comprendidas entre 192.168.22.2 y 192.168.168.22.9.
- Los servidores orientados al **almacenamiento de datos y la compartición de archivos en red** tendrán direcciones IP comprendidas entre 192.168.21.10 y 192.168.21.19.
- Los servidores destinados a la **ejecución de las máquinas virtuales de trabajo** por parte de los usuarios estarán localizados a partir de la IP 192.168.21.20.

En la Figura 3.3 se muestra la configuración actual de la red de datos.

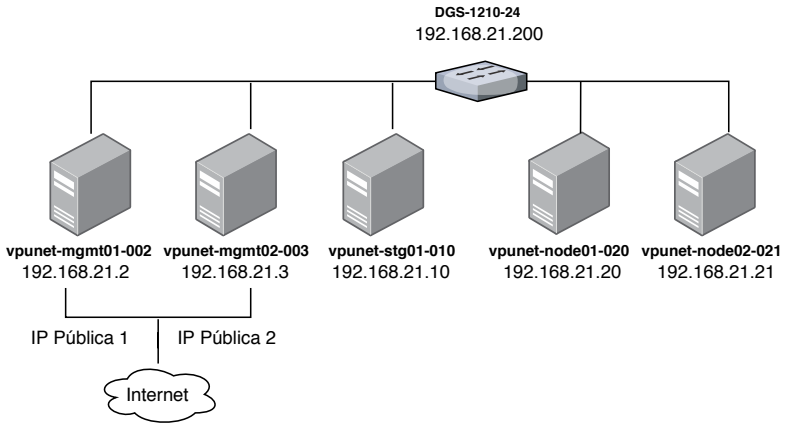


Figura 3.3: Diagrama de los componentes de la subred de datos.

3.3.3. Red de servicios

La red de servicios se compone de distintos contenedores específicos para una tarea. Estos contenedores se alojarán en alguno de los tres servidores localizados en la subred de gestión. Cada servicio poseerá una dirección IP en la subred 22, relacionada con el identificador del contenedor en el servidor de Proxmox correspondiente.

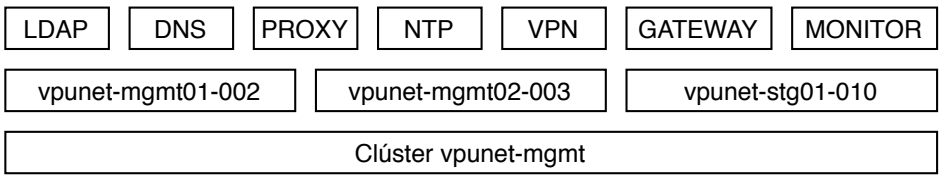


Figura 3.4: Diagrama de los componentes y contenedores que se sitúan en la red de servicios. El clúster está comprendido de tres servidores que a su vez alojan alguno de los servicios mostrados como contenedores.

Para esta red, etiquetada como subred 22, se requerirán los siguientes servicios:

- Una **puerta de enlace** que permita la interconexión entre las diferentes subredes previamente mencionadas. Este contenedor requerirá tantas interfaces como subredes se requieran.
- Un **servicio de resolución de nombres (DNS)** que permita resolver las IPs del laboratorio con los nombres de los equipos.

- Un **proxy** que habilite el tráfico de Internet a equipos o servicios que no tengan acceso directo a ello.
- Un servicio **LDAP** para efectuar un control de accesos a determinados servicios o máquinas virtuales. Además de esto se implementarán servicios adicionales que permitan a los usuarios y a los administradores gestionar sus credenciales y sus accesos de manera más directa a como se realiza la con la implementación por defecto.
- Un servicio de **NTP** que permita a todos los equipos mantener la misma hora de forma coordinada.
- Una **VPN** que permita el acceso remoto a la red de manera segura e individualizada.
- Un servicio de **monitorización** que registre los datos de los recursos de la red y que sea capaz de enviar alertas inmediatas en caso de cambios en los distintos sistemas.

3.3.4. Red de máquinas virtuales

En la red de máquinas virtuales estarán situados todos aquellas máquinas virtuales y contenedores destinados al uso para el trabajo en el laboratorio. Estas máquinas formarán parte de algún servidor orientado al procesamiento.

En la Figura 3.5 se muestra cómo se organizan las máquinas virtuales dentro de la red y cómo interaccionan las subredes para que las máquinas consigan acceso a Internet a través de un proxy.

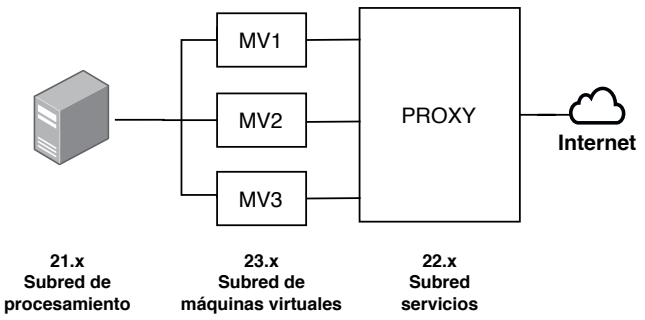


Figura 3.5: Ejemplo de interacción entre varias subredes para el acceso a internet de una máquina virtual mediante un proxy.

3.3.5. Vista global

Con toda la planificación de la red realizada, se presenta en la figura Figura 3.6 un diagrama de toda la red al completo, con todos los componentes mencionados anteriormente.

Los servidores de la red se separan a nivel de enlace en dos grupos aislados gracias a los dos switches instalados. La interconexión entre subredes se produce a través de los distintos elementos que están conectados al switch de datos. Para lograr una interacción transparente entre las distintas subredes se establecerá un servicio de puerta de enlace

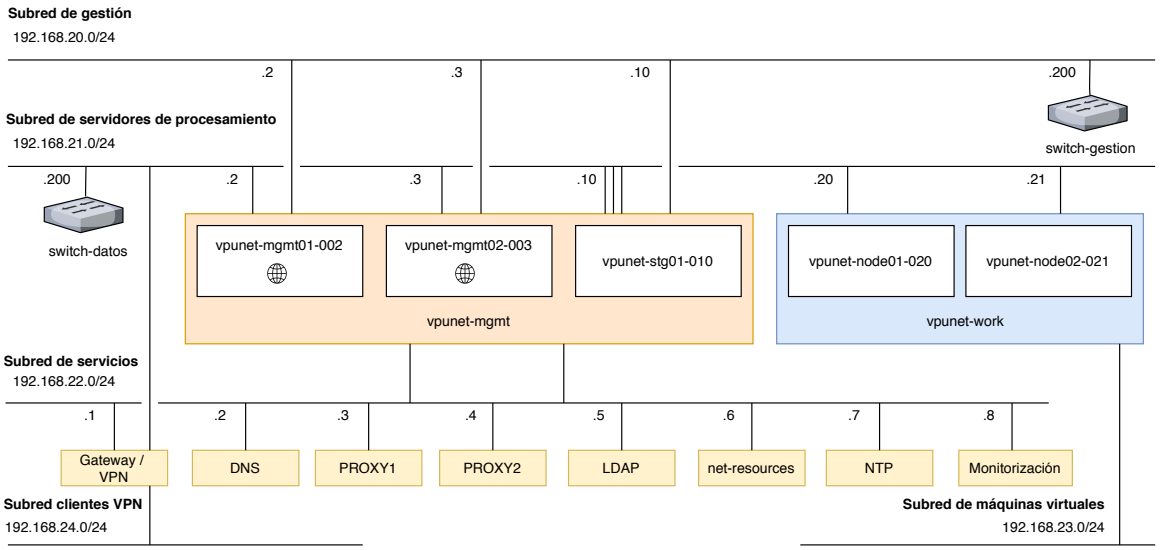


Figura 3.6: Diagrama general de todos los elementos de la red del laboratorio, incluyendo servidores, switches, clústers, servicios y subredes.

que tendrá una IP asignada en cada subred que se quiera interconectar, permitiendo redireccionar los paquetes entre una subred y otra.

En la red, hay multitud de componentes que no tienen un acceso directo al exterior, un proxy permitirá a esas máquinas contactar con Internet cuando sea necesario. De la misma forma, el acceso desde cualquier equipo conectado a Internet al interior de la red, se otorgará a través de un túnel VPN que requerirá que los usuarios estén correctamente autentificados. Además de esto, será posible redireccionar algunos servicios para que sean accesibles desde Internet sin necesidad de la utilización de un servicio adicional.

Capítulo 4

Implementación

4.1. Introducción

En este capítulo se explicará de qué forma ha sido posible inicializar los servidores para garantizar su interconexión y garantizar que se encuentren preparados para la ejecución y el respaldo de contenedores y máquinas virtuales.

4.2. Inicialización de servidores

Para dar de alta un servidor será necesario una unidad USB donde copiar el sistema a instalar, en nuestro caso, Proxmox 6.1.

Tras acceder a página de descarga de Proxmox[1] y obtener la ISO de Proxmox VE 6.1, se inicializará una unidad USB con la imagen ISO descargada utilizando la herramienta *dd*. Se ubicará el USB con *fdisk* o *lsblk*, que en este caso corresponderá con */dev/sdb*:

```
# dd if=proxmox-ve_6.1-1.iso of=/dev/sdb status="progress"
1589692+0 registros leidos
1589692+0 registros escritos
813922304 bytes (814 MB, 776 MiB) copied, 115,963 s, 7,0 MB/s
```

4.2.1. Proceso de instalación

Tras inicializar el USB se arrancará con él mismo el servidor objetivo y se seguirán los pasos de la instalación de Proxmox:

1. La **configuración de los discos** será diferente dependiendo del tipo de máquina:

    1.1. Existen dos tipos de **servidores de gestión**: el primer tipo corresponde a los servidores orientados exclusivamente para servicios y el segundo a los servidores de almacenamiento, que también pueden albergar servicios relacionados.

        1.1.1. Los servidores que se utilicen **exclusivamente para servicios** cuentan con dos unidades de estado sólido SSD. Se utilizará para la instalación

[1] http://download.proxmox.com/iso/proxmox-ve_6.1-2.iso

el sistema de archivos ZFS bajo RAID1[2] para las dos unidades SSD *sda1 y sda2*

1.1.2. El **servidor de discos** cuenta con una configuración especial, la unidad principal de almacenamiento principal es una unidad flash de 14GB. Como esta máquina se ubicará en el mismo clúster que en las otras dos, habrá que adaptar la configuración de almacenamiento para que sea simétrica. En el momento de la instalación se formateará con el sistema de archivos *EXT4* la partición donde residirá el sistema operativo.

Los **servidores de procesamiento o de trabajo** cuentan con tres discos: dos de ellos unidades SSD y un disco mecánico HDD. El disco del sistema corresponderá al SSD. Para el SSD de instalación se usará el sistema de archivos *EXT4*.

2. La **configuración de la red** de los equipos será como la siguiente:

   2.1. Para los **servidores de gestión** se utilizará la primera interfaz con una IP pública con acceso a Internet.

   2.2. Para los **servidores de procesamiento** se utilizará la primera interfaz con una IP local asociada a la red de datos del laboratorio.

3. Los pasos de configuración de zona horaria y de contraseña de usuario root serán los mismos para todos los servidores.

4.2.2. Postinstalación

Para que las máquinas y los servicios funcionen correctamente, será necesario añadir interfaces de red en Proxmox que habiliten los accesos a las redes necesarias. Además se añadirá la dirección IP de las puertas de enlace que se implementarán posteriormente para efectuar la interconexión de las subredes. En la Tabla 4.1 se detalla la configuración para cada tipo de servidor instalado.

	Interfaz	IP	Puerta de enlace	Descripción
vpunet-mgmt01-002	vmbr0	IP pública 1	IP Gateway UAM	Acceso directo a internet
	vmbr1	192.168.20.2	192.168.20.1	Enlace a switch de gestión
	vmbr2	192.168.21.2	192.168.21.1	Enlace a switch de datos
	vmbr3	10.0.0.1	-	Bridge de acceso directo a internet para MVs
vpunet-mgmt02-003	vmbr0	IP pública 2	IP Gateway UAM	Acceso directo a internet
	vmbr1	192.168.20.3	192.168.20.1	Enlace a switch de gestión
	vmbr2	192.168.21.3	192.168.21.1	Enlace a switch de datos
	vmbr3	10.0.0.1	-	Bridge de acceso directo a internet para MVs
vpunet-stg01-010	vmbr1	192.168.20.10	192.168.20.1	Enlace a switch de gestión
	vmbr2	192.168.21.10	192.168.21.1	Enlace a switch de datos
vpunet-node01-020	vmbr0	192.168.21.20	192.168.21.1	Enlace a switch de datos

Tabla 4.1: Esquema de configuración de red de los servidores instalados

Para los servidores con acceso a Internet se habilitará el **redireccionamiento de paquetes** para poder hacer públicos servicios disponibles en las máquinas virtuales y contenedores:

[2]RAID1 indica que la mitad del almacenamiento se usará con una copia de los datos, con esto se garantiza la redundancia de discos y la integridad de los datos.

- Para que el redireccionamiento persista entre reinicios, se añadirá la línea `net.ipv4.ip_forward = 1` en el fichero `/etc/sysctl.conf`. Es recomendable reiniciar el servidor para asegurar que la opción ha sido activada.

- Se añadirá la regla correspondiente para el redireccionamiento en *iptables*, las tablas del cortafuegos del kernel de Linux:

```
# iptables -t nat -A POSTROUTING -j MASQUERADE
```

- Se instalarán los paquetes necesarios para hacer persistentes las reglas de redireccionamiento que se añadan en un futuro:

```
# apt install iptables-persistent netfilter-persistent
```

- Ejecutando `iptables-save >/etc/iptables/rules.v4` se realizará una copia de la configuración actual, editando el archivo se redireccionarán los puertos de las máquinas virtuales a la IP pública de la máquina especificada. Por ejemplo, para redireccionar el puerto TCP 80 de un contenedor al puerto 8080 de la IP pública de un servidor con acceso a Internet se añadirá al fichero `rules.v4` una regla como la siguiente:

```
-A PREROUTING -p tcp --dport 8080 -j DNAT --to-destination
   192.168.22.213:80
```

  Reiniciando el servicio con `systemctl restart netfilter-persistent` se aplicarán los cambios y el servicio quedará accesible a través de la IP pública.

Existen contenedores que necesitan un acceso directo a Internet, es decir, que no necesiten Proxys ni VPNs para ello. Se habrá creado una interfaz auxiliar *vmbr3* que habrá que añadir en todas aquellas máquinas virtuales o contenedores que lo requieran, asignándoles una IP en esa interfaz en el rango 10.0.0.x y utilizando 10.0.0.1 como puerta de enlace. Los detalles de esta interfaz se pueden visualizar en la Tabla 4.1. Para que el acceso funcione correctamente, habrá que configurar el servidor en el que reside Proxmox añadiendo la regla `-A POSTROUTING -s 10.0.0.0/24 -o vmbr0 -j MASQUERADE` al fichero `/etc/iptables/rules.v4`. De esta manera, `vmbr3` enrutará las peticiones de IPs externas a la interfaz que usa el servidor Proxmox para acceso a internet, `vmbr0`.

4.2.3. Configuración del clúster de gestión

Como se muestra en la Figura 4.1, los servidores de gestión deben de configurarse como nodos en un clúster. El clúster tiene como objetivo facilitar la interacción entre máquinas y la migración de servicios entre ellas en caso de fallo. Para dar de alta el clúster se seguirán los siguientes pasos:

- Antes de empezar, revisar el fichero `/etc/hosts` de todas las máquinas implicadas para que la dirección IP de gestión de Proxmox coincida con la subred asignada al switch de gestión. Por ejemplo, para la segunda máquina:

```
root@vpunet-mgmt02-003:~# cat /etc/hosts
127.0.0.1 localhost.localdomain localhost
192.168.20.3 vpunet-mgmt02-003.vpu.eps.uam.es vpunet-mgmt02-003
...
```

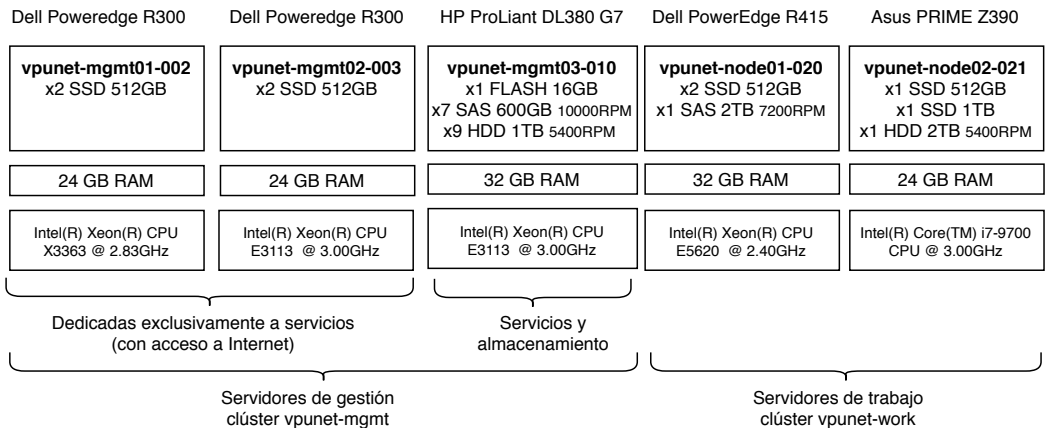


Figura 4.1: Especificaciones técnicas de las máquinas instaladas, acompañadas de su configuración de discos y su cometido en la red.

- Es necesario elegir cuál va a ser el nodo maestro, en este caso *vpunet-mgmt01-002*. Tras acceder a él se creará el clúster con `pvecm create vpunet-mgmt`.
- Acceder a los nodos restantes. Por cada nodo que se quiera añadir, iniciar la adhesión al nodo maestro con el comando `pvecm add 192.168.20.2` y reiniciar cada nodo antes de añadir otro, comprobando que `systemctl status corosync.service` no reporta ningún error.
- Abriendo el fichero `/etc/pve/.members` en cualquiera de los nodos, se efecuará la comprobación de que las direcciones IP corresponden a la interfaz correcta:

```
root@vpunet-stg01-010:~# cat /etc/pve/.members
{
"nodename": "vpunet-stg01-010",
"version": 17,
"cluster": { "name": "vpunet-mgmt-003", "version": 8, "nodes": 3,
    "quorate": 1 },
"nodelist": {
  "vpunet-mgmt01-002": { "id": 1, "online": 1, "ip":
     "192.168.20.2"},
  "vpunet-mgmt02-003": { "id": 2, "online": 1, "ip":
     "192.168.20.3"},
  "vpunet-stg01-010": { "id": 3, "online": 1, "ip":
     "192.168.20.10"}
  }
}
```

4.3. Instanciación de servicios

Como ya se ha comentado antes, los servicios necesarios de la red se implementarán como contenedores LXC, esto es beneficioso para no centralizar los servicios en un único punto de fallo y la posibilidad de aislar unos servicios de otros, facilitando la realización de copias de seguridad. La mayoría de estos contenedores se basarán en el sistema operativo Debian [4], un sistema diseñado para ser muy estable y ampliamente utilizado en servidores.

4.3.1. Puerta de enlace

Una puerta de enlace es un elemento de red que tiene acceso a distintas interfaces de red y permite la comunicación de distintas subredes.

Para su creación, se va a partir del contenedor de VPN creado anteriormente para que la conectividad de los clientes de la VPN con la subred sea satisfactoria, aunque la puerta de enlace será utilizada también en todas las máquinas que lo requieran.

En la configuración del contenedor en Proxmox se añadirán tantas interfaces como subredes a direccionar, tal y como se muestra en la Figura 4.2:

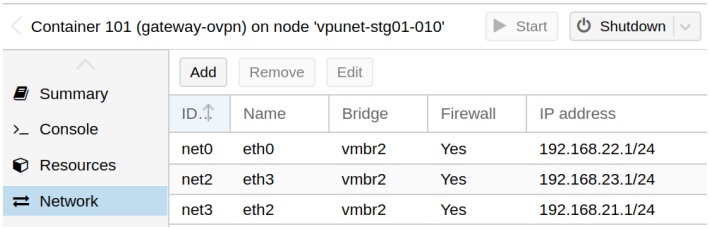


Figura 4.2: Configuración de las interfaces de red en el contenedor de la puerta de enlace

Para que el contenedor pueda dirigir peticiones hacia otras subredes, será necesario habilitar el redireccionamiento de paquetes IP a nivel de núcleo añadiendo la línea *net.ipv4.ip_forward = 1* en `/etc/sysctl.conf`. También se añadirá una regla en la tabla NAT del sistema con *iptables -t nat -A POSTROUTING -j MASQUERADE*.

Se instalará el paquete *iptables-persistent* que permitirá guardar los cambios de forma permanente con el comando *iptables-save > /etc/iptables/rules.v4*.

Una vez hecho esto será posible conectar cualquier máquina con otra máquina que esté en una subred distinta. Si se quisiera limitar el acceso entre redes específicas se podrán añadir las reglas correspondientes en el fichero *rules.v4*.

4.3.2. Proxy

Un proxy [5] hace de intermediario entre las peticiones de dos máquinas para un tipo de tráfico determinado. Se utilizará un proxy para aquellas máquinas que no tengan acceso directo a Internet pero necesiten una conexión puntual. Para ello, se creará un contenedor vacío basado en Debian 9 al que se instalará *squid*:

```
# apt install squid apache2-utils
```

Acto seguido se reemplazará la configuración de `/etc/squid/squid.conf` con lo siguiente:

```
http_port 8080
visible_hostname VPUProxy
forwarded_for off
access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache deny all
dns_nameservers 8.8.8.8 8.8.4.4
positive_dns_ttl 5 minutes #positive response
negative_ttl 5 minutes #erroneous response
shutdown_lifetime 0 seconds
auth_param basic children 5
```

```
auth_param basic realm Squid Basic Authentication
auth_param basic credentialsttl 5 hours
http_access allow password
```

Una vez reiniciado el servicio con `systemctl restart squid.service` será posible utilizarlo en cualquier máquina que lo requiera, utilizando la IP de la máquina con el puerto 8080.

Si se desea establecer una autentificación de usuario y contraseña habrá que utilizar el siguiente comando:

```
        # htpasswd -c /etc/squid/passwd <username>
```

4.3.3. DNS

Un servidor DNS permite resolver nombres de máquinas en direcciones IP. Para ello, se instalará en un contenedor con Debian 9 los paquetes *pdns-recursor* y *dnsutils*. Para la configuración del servidor DNS sólo habrá que estar seguros de que la configuración en el fichero `/etc/powerdns/recursor.conf` contiene estas líneas:

```
local-address=0.0.0.0
export-etc-hosts=on
```

Una vez reiniciado el servicio *pdns-recursor* podremos añadir la relación entre IP y nombres en el archivo `/etc/hosts`.

Para configurar un cliente solamente habrá que añadir las siguientes líneas al fichero `/etc/resolv.conf`:

```
search vpu.eps.uam.es
nameserver 192.168.22.2
```

4.3.4. NTP

Un servicio NTP permite mantener el tiempo sincronizado a todos los clientes que se conecten. En nuestras circunstancias, va a ser habitual que muchas máquinas no cuenten con acceso directo a Internet, por lo que un contenedor con acceso a Internet, obtendrá la hora de un *pool* remoto y lo redistribuirá a la red local. En la Figura 4.3 se detalla su funcionamiento.

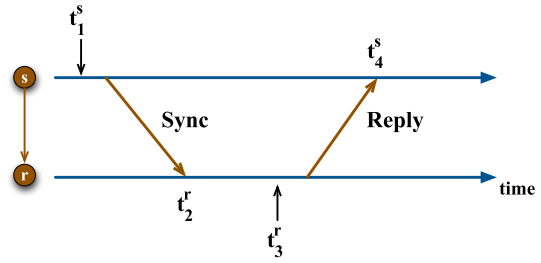


Figura 4.3: Diagrama de sincronización NTP. Fuente: [6]. En t_1, el cliente envía una petición de sincronización que es recibida en el instante t_2 al servidor. El servidor envía su tiempo en t_3 y es recibido por el cliente en t_4, que realiza ajustes graduales con el retardo en las peticiones hasta ajustarse al tiempo del servidor.

En un contenedor con Debian 10, se instalará el paquete *ntp*. Por defecto el servicio estará configurado y podrá distribuir la hora a las demás máquinas de la red.

Con `ntpq -p` se podrá comprobar con qué servidores sincroniza el servidor.

En los clientes se instalará también el paquete *ntp* y se configurará `/etc/ntp.conf` reemplazando las líneas que empiecen por *pool* por una línea que apunte al servidor ntp local: *server ntp.vpu.eps.uam.es*.

4.3.5. Control de acceso de usuarios a máquinas virtuales

Es necesario también un método para identificar usuarios y gestionar permisos. Para ello se usará *OpenLDAP* [7], una implementación de código abierto para LDAP. Se utilizará una plantilla *debian-9-turnkey-openldap_15.1-1* que incluirá el servicio *OpenLDAP* y la interfaz de administración *phpLDAPAdmin*. Nada más arrancar el contenedor se nos pedirán ciertos detalles de instalación como la contraseña de administrador y el identificador base (*basedn*) que en nuestro caso estableceremos a *dc=vpu,dc=eps,dc=uam,dc=es*. La organización final de usuarios y grupos será como la que se muestra en la Figura 4.4.

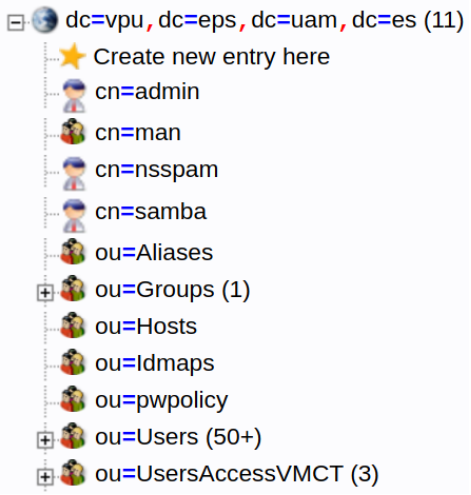


Figura 4.4: Árbol de las distintas entidades del servicio *OpenLDAP*. En él, se pueden visualizar los grupos de usuarios *(ou=Users)*, administración *(cn=admin)* y diversos grupos para habilitar el acceso a máquinas virtuales específicas *(ou=UsersAccessVMCT)*.

Al acceder a la dirección IP de la máquina configurada con un navegador se mostrará una interfaz de administrador en la que se podrán añadir grupos y usuarios. Para los usuarios del laboratorio se usará una plantilla XML modificada con campos personalizados.

Para habilitar el inicio de sesión por *LDAP* en máquinas basadas en *Ubuntu* serán necesarios varios pasos:

- Instalar los paquetes necesarios:

```
# apt install libnss-ldap libpam-ldap ldap-utils nscd
```

- Cambiar la configuración de `/etc/ldap.conf` a las siguientes credenciales:

```
base dc=vpu,dc=eps,dc=uam,dc=es
uri ldap://192.168.22.5
...
ldap_version 3
...
pam_passwd exop
...
nss_base_passwd ou=Users,dc=vpu,dc=eps,dc=uam,dc=es
nss_base_shadow ou=Users,dc=vpu,dc=eps,dc=uam,dc=es
nss_base_group  ou=Groups,dc=vpu,dc=eps,dc=uam,dc=es
```

- Habilitar LDAP en el fichero `/etc/nsswitch.conf`

```
passwd:         compat ldap
group:          compat ldap
shadow:         compat ldap
hosts:          files mdns4_minimal [NOTFOUND=return] dns ldap
```

- Configurar la auntentificación con LDAP

```
sudo auth-client-config -t nss -p lac_ldap
sudo pam-auth-update
```

- Verificar que los ficheros *common-account, common-auth, common-password y common-session* han sido configurados con LDAP.

Una vez hecho esto se probará si se identifica a un usuario de prueba configurado a través de el servidor LDAP:

```
# getent passwd prueba
prueba:*:2001:100:prueba prueba:/home/prueba:/bin/bash
```

4.3.6. OpenVPN

Un servicio VPN permite el acceso a redes remotas de forma segura, para ello, se establece un servidor VPN en la red a la que se quiera acceder y se configurará un cliente en una ubicación remota (Figura 4.5).

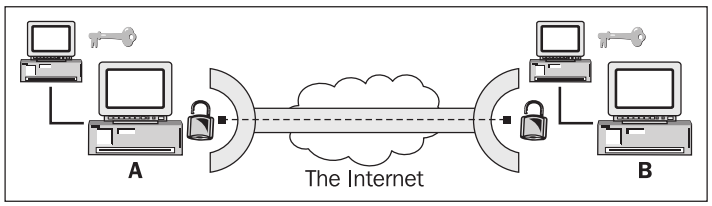


Figura 4.5: Diagrama de conexionado de dos equipos en subredes distintas que a través de un túnel cifrado, pueden compartir información como si estuviesen dentro de la misma red física. (Fuente: [8])

Para la creación del servicio VPN se creará un contenedor LXC con la plantilla *debian-9-turnkey-openvpn_15.1-1*, que incluye los scripts y paquetes necesarios para su funcionamiento. Tras definir en la instalación que la subred que se va a utilizar

para los clientes de la VPN será la 24, el instalador tratará de generar las claves necesarias. Estas credenciales habrá que actualizarlas mediante la edición del fichero `etc/openvpn/easy-rsa/vars` y regenerando todos los certificados del servidor mediante la siguiente secuencia de comandos:

```
cd /etc/openvpn/easy-rsa
./cleanall
./build-ca
./build-dh
mkdir -p /etc/openvpn/easy-rsa/keys/crl.jail//etc/openvpn/server.ccd
mkdir -p /etc/openvpn/easy-rsa/keys/crl.jail/tmp
openvpn --genkey --secret keys/ta.key
./build-key-server server
```

Se puede añadir un usuario de prueba mediante el comando `openvpn-addclient name mail` que generará un fichero *.ovpn* único que contendrá todas las claves listas para su distribución final.

El acceso a las distintas subredes se puede controlar editando el fichero *server.conf* y añadiendo la línea *push route 192.168.X.X 255.255.255.0"* especificando la subred que se quiera añadir.

Por, último, si se desea integrar la autenticación de usuarios por LDAP en el servicio VPN, se instalará el paquete *openvpn-auth-ldap* y se editará el fichero */etc/openvpn/auth/auth-ldap.conf* para contactar con el servidor de usuarios. Acto seguido se configurará el servidor para que pida las credenciales del servidor de usuarios:

```
# cat /etc/openvpn/server.conf
...
plugin /usr/lib/openvpn/openvpn-auth-ldap.so /etc/openvpn/auth/auth-
    ldap.conf
client-cert-not-required
```

4.3.7. Monitorización

Para realizar un seguimiento de todos los componentes de la red, se instalará en un contenedor con Debian 10, un servidor *Check_MK*. Check_MK es una herramienta de monitorización extensible con complementos adicionales y que en su versión de código abierto está basada en *Nagios* [9].

4.3.7.1. Instalación y puesta en marcha del servicio de monitorización

Se descargará y se instalará el .deb desde la página oficial[3]. Se creará un nuevo sitio con el comando *omd create "nombre_sitio"*, al realizar esto se generará una contraseña para el usuario *cmkadmin*. Si todo se ha configurado bien, se podrá acceder desde el navegador al panel de monitorización que se muestra en la Figura 4.6 a través de la dirección IP seguida del nombre del sitio previamente configurado.

4.3.7.2. Instalación del agente de monitorización

Para añadir equipos o servicios para monitorizar, se necesitará instalar un agente en el nuevo equipo que transmitirá la información necesaria al servidor Check_MK. Para

[3]Infrastructure & Application Monitoring *https://checkmk.com* (Visitado: 15 Abril 2020)

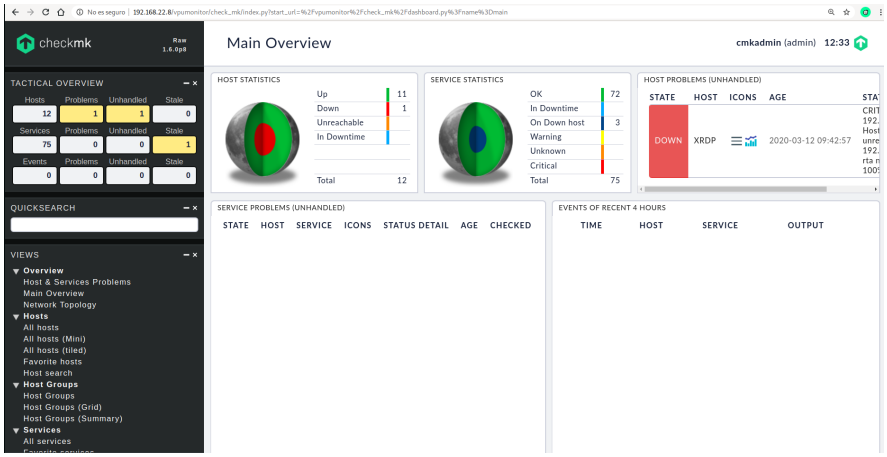


Figura 4.6: Panel de control del servicio de monitorización de *Check_MK*

que máquinas basadas en Debian sean monitorizables, habrá que dirigirse a la sección *Monitoring Agents* de la barra lateral y descargar el agente .deb que habrá que instalar en la máquina que se va a añadir. Finalmente, sólo habrá que seleccionar *Hosts* en la barra lateral para añadir un nuevo equipo y seleccionar los servicios a los que se les quiera hacer un seguimiento.

4.3.7.3. Organización de equipos

Para agregar un nuevo equipo para monitorizar a la interfaz, habrá que situarse en *Hosts* en la barra lateral y a *Add Host*, tras introducir la dirección IP correspondiente y pasar al siguente apartado, se mostrarán los servicios disponibles y se tendrá la posibilidad de elegir cuáles de ellos se quieren monitorizar.

La organización de la monitorización de todos los equipos se realizará en tres carpetas: una destinada a servidores físicos, otra a servicios esenciales de la red y por último otro directorio destinado a las máquinas virtuales del sistema. Esta organización permitirá, más adelante, crear reglas específicas para cada tipo de servidor. Para crear todas estas carpetas, simplemente habrá que dirigirse a *Hosts* en la barra lateral y a *New Folder*.

4.3.7.4. Notificaciones por correo electrónico

Es interesante contar con un sistema que pueda lanzar una notificación cuando algún componente de la red deje de estar disponible o cambien algunos de sus parámetros.

Para notificar por correo, es necesario configurar un servidor de correo como postfix en la máquina de Check_MK.

- Se instalarán los paquetes necesarios:

```
# apt-get install postfix mailutils libsasl2-2 ca-certificates
   libsasl2-modules
```

- Se configurará *postfix* con *Gmail*:

```
# cat /etc/postfix/main.cf
relayhost = [smtp.gmail.com]:587
relayhost = [smtp.gmail.com]:587
```

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
smtp_use_tls = yes
```

- Se configurarán los parámetros de usuario de correo:

```
# cat /etc/postfix/sasl_passwd
[smtp.gmail.com]:587    monitor@gmail.com:<password>
# postmap /etc/postfix/sasl_passwd
# service postfix restart
# echo "This is message body" | mail -s "This is Subject"
   monitor@gmail.com
```

Si todo se ha configurado correctamente, se recibirá un correo de prueba y por defecto Check_MK enviará notificaciones cuando el estado cambie.

4.3.7.5. Notificaciones por Telegram

Otra forma posible de notificar errores es mediante el servicio de mensajería *Telegram*. Se utilizará un script[4] que habrá que modificar. Para ello se creará un bot de *Telegram* y seguidamente se procederá a la extracción de su *API_KEY* y *CHAT_ID*. Cambiando las variables correspondientes en el script, ubicándolo en */local/share/check_mk/notifications* y otorgándole permisos de ejecución. Por último queda ir a la web de Check_MK, navegar a *Notifications* y añadir una nueva regla, seleccionando *telegram* como método de notificación.

4.4. Almacenamiento distribuido

4.4.1. Unidades de red

Para hacer copias de seguridad remotas, se habilitará el servidor de discos *vpunet-stg01-010* compartiendo ciertos directorios a la red. Para ello se utilizará ZFS [10] como herramienta para organizar las unidades de disco. Se crearán dos *pools* de discos. Un *pool* es utilizado en ZFS para representar un conjunto de discos en el cual los datos se extienden por todos ellos. En este caso, nuestros *pools* será de tipo *RAID-Z2*, que garantiza una tolerancia a fallos de dos discos.

4.4.1.1. Creación de los sistemas de archivos

- Se creará un *pool* con el nombre *local-zfs*, que se utilizará para el almacenamiento de las máquinas en ejecución y permitirá que el servidor de discos tenga la misma configuración básica de almacenamiento que los otros dos servidores.

```
# zpool create rpool /dev/sda
# zfs create rpool/data
```

[4]Bot Telegram para recibir notificaciones de Check_mk *https://www.evaristorivieccio.es/2019/04/bot-telegram-para-recibir.html* (Visitado: 20 Abril 2020)

- Se creará un *pool* llamado *stg-pool1*, compuesto por todas las unidades de almacenamiento SAS menos la utilizada por *local-zfs*.

```
# zpool create -f -o ashift=12 stg-pool1 raidz2 /dev/sdb /dev/sdc
    /dev/sdd dev/sdg /dev/sdh /dev/sdm
# zfs set mountpoint=/mnt/stg-pool1 stg-pool1
```

- Se creará otro *pool* denominado *stg-pool2*, compuesto por todas las unidades de disco mecánicas.

```
# zpool create -f -o ashift=12 stg-pool2 raidz2 /dev/sde /dev/sdf
    /dev/sdi /dev/sdj /dev/sdk /dev/sdl /dev/sdn /dev/sdo /dev/
   sdp
# zfs set mountpoint=/mnt/stg-pool2 stg-pool2
```

- Con el comando `zpool list` se muestra cuánto almacenamiento hay disponible finalmente para cada `pool` creado:

```
root@vpunet-stg01-010:~# zpool list
NAME         SIZE  ALLOC   FREE      FRAG  CAP  DEDUP  HEALTH
stg-pool2   8.17T   380G  7.80T  ...   0%   4%  1.00x  ONLINE
stg-pool1   3.27T   151G  3.12T  ...   0%   4%  1.00x  ONLINE
rpool        556G  10.1G   546G  ...   0%   1%  1.00x  ONLINE
```

4.4.1.2. Estructura de directorios

Se organizarán los distintos directorios de compartición mediante *datasets*[5] creados con el comando `zfs create <punto_montaje>`. En el pool llamado *stg-pool2* se establecerán los almacenamientos que utilizará Proxmox para todas sus máquinas, contendrá subdirectorios para almacenar backups y plantillas de contenedores. El pool llamado *stg-pool1* será orientado a los directorios para la compartición de archivos entre usuarios:

- El dataset /home se utilizará como directorio personal para los usuarios.
- El dataset /intercambio es un espacio temporal común para la compartición entre usuarios.
- /man es un directorio reservado a los administradores del sistema.
- /utils contiene diversas herramientas y aplicaciones necesarias para el trabajo en el laboratorio.

El pool *rpool* se utilizará para las máquinas locales que se ejecuten en Proxmox.

4.4.1.3. Compartición en red mediante NFS

Para compartir directorios en red, se instalará el módulo NFS en el servidor con el paquete `nfs-kernel-server`. Con esto, ya se podrán publicar los directorios a la red mediante la edición del fichero `/etc/exports`:

[5]Un dataset de ZFS es una división lógica de un pool de discos, representado en el sistema de archivos como un directorio más al que se le pueden otorgar atributos especiales como cuotas.

```
/mnt/stg-pool2/iso-templates    *(rw,sync,no_root_squash)
/mnt/stg-pool2/vm-backups       *(rw,sync,no_root_squash)
/mnt/stg-pool2/services-backups *(rw,sync,no_root_squash)
/mnt/stg-pool1/home            *(rw,sync,no_root_squash)
/mnt/stg-pool1/intercambio     *(rw,sync,no_root_squash)
/mnt/stg-pool1/utils           *(rw,sync,no_root_squash)
```

Se ha configurado en el servidor de discos el acceso por LDAP, procedimiento previamente descrito en la Sección 4.3.5. De esta manera, cada vez que se agregue un usuario, se podrá limitar el espacio del directorio personal del mismo mediante el comando *zfs set userquota@usuario=2G stg-pool1/home*. Los demás datasets no tienen ninguna cuota asignada, aunque en *intercambio* se usa *tmpreaper* para borrar su contenido cada siete días.

Para que las unidades de plantillas y backups puedan ser utilizadas dentro de la interfaz de Proxmox independientemente del nodo habrá que añadir estas unidades NFS en el fichero `/etc/pve/storage.cfg` de cada servidor Proxmox:

```
nfs: nfs-user-backups
        export /mnt/stg-pool2/vm-backups
        path /mnt/pve/nfs-user-backups
        server 192.168.21.10
        content backup,iso,vztmpl
        maxfiles 3
        options vers=4

nfs: nfs-services-backups
        export /mnt/stg-pool2/services-backups
        path /mnt/pve/nfs-services-backups
        server 192.168.21.10
        content backup
        maxfiles 3
        options vers=4

nfs: nfs-iso-templates
        export /mnt/stg-pool2/iso-templates
        path /mnt/pve/nfs-iso-templates
        server 192.168.21.10
        content images,backup,iso,vztmpl,rootdir
        maxfiles 3
        options vers=4
```

Una vez hecho esto será posible acceder a las unidades remotas y realizar copias de seguridad periódicas en ellas.

4.4.2. Servicios auxiliares de almacenamiento en red

Para facilitar y centralizar el acceso al almacenamiento distribuido con un control de acceso de usuarios se creará un contenedor basado en Debian 10. En este contenedor se montarán las unidades NFS del servidor de discos y se instalarán los servicios *Samba*[6] y *SFTP* [7].

[6] Samba es una la implementación para sistemas UNIX del sistema de archivos compartidos de *Windows*.

[7] *SSH File Transfer Protocol* es una implementación para sistemas basados en UNIX de un sistema de archivos compartido cifrado.

Para montar las unidades de NFS necesarias primero se instalará *nfs-common* para seguidamente modificar el fichero */etc/fstab*. Se añadirá una unidad común a todos los usuarios y un directorio remoto *home* que contendrá los directorios personales de cada usuario.

```
# cat /etc/fstab
192.168.21.10:/mnt/stg-pool1/intercambio /mnt/intercambio nfs
   defaults 0 0
192.168.21.10:/mnt/stg-pool1/home        /home            nfs
   defaults 0 0
```

Primero se instalará samba en la máquina:

```
# apt install samba samba-common
```

Se añadirá un usuario que utilizarán los clientes de Samba para efectuar las operaciones de archivos y se establecerá como propietario ese usuario para los directorios a compartir.

```
# adduser --system shareuser
# chown -R shareuser /mnt/intercambio
```

A continuación se publicará ese directorio a Samba añadiendo las siguientes líneas a `/etc/samba/smb.conf`:

```
[intercambio]
  comment = Disco Comun
  writable = yes
  browseable = yes
  path = /mnt/intercambio
  public = yes
  create mask = 0644
  directory mask = 0755
  force user = shareuser
```

Adicionalmente, se requiere un acceso controlado a los directorios */home* de cada usuario, por lo que se requerirá configurar el acceso de clientes LDAP al sistema tal y como se mostró en la Sección 4.3.5. Además de ello habrá que realizar una pequeña modificación en el fichero que se muestra a continuación. Con esto se consigue que cada directorio de usuario que sea creado sea privado y sólo pueda ser visto por el propio usuario:

```
# cat /etc/pam.d/common-session
...
session required pam_mkhomedir.so skel=/etc/skel umask=077
...
```

4.5. Contenedor con acceso remoto multiusuario

4.5.1. Instalación de escritorio multiusuario

Para que los usuarios puedan utilizar con comodidad los servidores de procesamiento, se inicializará un contenedor basado en *Ubuntu* 18.04.4 integrando en él un entorno de escritorio basado en LXDE. Para el acceso remoto se utilizará la implementación

libre de RDP[8], XRDP. Primero se instalarán los paquetes necesarios tras conectar el contenedor a Internet mediante proxy o conexión directa.

```
# apt install xrdp xorgxrdp tasksel
# apt install lubuntu-desktop
```

Se editará el fichero *startwm.sh* para ajustar sus últimas líneas. El objetivo es que todos los usuarios que inicien sesión por RDP lancen una sesión de *Lubuntu*.

```
# cat /etc/xrdp/startwm.sh
# test -x /etc/X11/Xsession && exec /etc/X11/Xsession
# exec /bin/sh /etc/X11/Xsession
exec /usr/bin/lxsession -s Lubuntu -e LXDE
```

4.5.2. Control de acceso mediante LDAP

Para configurar el acceso al contenedor con los usuarios de LDAP se seguirá el proceso descrito en la Sección 4.3.5. Una funcionalidad adicional que se ha implementado en este proceso, es otorgar el acceso sólo a ciertos usuarios que se encuentren en grupos específicos de LDAP. Para ello se instalará el paquete *libpam-ldapd* y se modificará el fichero */etc/nslcd.conf* para añadir la siguiente línea:

```
pam_authc_search (&(memberUid=$username)(cn=UsersAccessVMCT_110))
```

Con esto se permite el acceso a todos los usuarios que formen parte del grupo auxiliar *cn=UsersAccessVMCT_110* donde 110 es el identificador del contenedor instalado. Este proceso se puede extrapolar a diferentes máquinas para distintos tipos de usuarios.

4.5.3. Configuración de las cuotas

Una característica adicional de este contenedor es la limitación de uso de espacio en disco por usuario, para ello, habrá que configurar el contenedor para que soporte cuotas:

- En la configuración de Proxmox del contenedor se habilitará la opción de cuotas en la sección de disco (Figura 4.7).

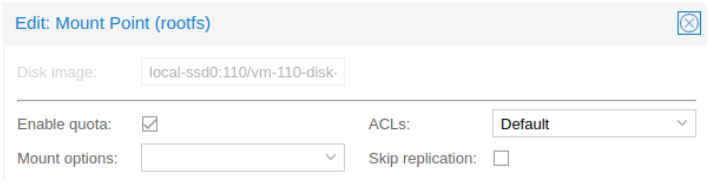


Figura 4.7: Habilitando cuotas en las opciones de disco de un contenedor LXC.

- Se instalarán los paquetes de cuota y se habilitarán para el sistema de archivos actual:

```
touch /aquota.user /aquota.group
chmod 0600 /aquota.*
quotacheck -cmug /
quotaon -avug
```

[8]RDP (Remote Desktop Protocol) es un protocolo propietario desarrollado por Microsoft. RDP utiliza por defecto el puerto 3389 para efectuar sus conexiones.

- Para que automáticamente se le asigne al usuario una cuota de alerta de 18GB y una cuota máxima de 20GB, será necesario realizar un script que se ejecute cada vez que un usuario se conecte:

```
root@XRDP:~# cat /opt/setuserquota.sh
#!/bin/sh
quotatool -u $PAM_USER -b -q 18G -l 20G /
```

  Una vez hecho esto, se añadirá una regla al administrador de sesiones de XRDP para ejecutar el script en cada inicio de sesión.

```
# cat /etc/pam.d/xrdp-sessman
session required pam_exec.so /opt/setuserquota.sh $PAM_USER
```

- Por último se realizará una unidad de *SystemD* para que al arrancar el contenedor se habilite automáticamente el sistema de cuotas:

```
root@XRDP:/etc/systemd/system# cat enable_quota.service
[Unit]
Description=Enable user and group quotas

[Service]
Type=oneshot
ExecStart=/sbin/quotaon -avug
RemainAfterExit=true
[Install]
WantedBy=multi-user.target
root@XRDP:# systemctl enable enable_quota.service
```

Al realizar todos los procedimientos y ejecutar un cliente de RDP con la dirección del contenedor y un usuario válido, se iniciará la sesión de escritorio remoto.

4.6. Formularios simplificados para la gestión de usuarios

Para que los usuarios puedan gestionar sus credenciales de manera sencilla, se ha instanciado un contenedor que consiste en un formulario básico para el cambio de contraseña (Figura 4.8), utilizando el paquete *ldap-passwd-webui-waitress* bajo Alpine Linux. Además, se ha implementado en otro contenedor una instancia de *LDAPCherry* (Figura 4.9), que permite a los administradores añadir nuevos usuarios de manera más directa que con la interfaz por defecto del contenedor LDAP.

Tras la instalación de LDAPCherry con *git clone https://github.com/kakwa/ldapcherry*, se usarán los siguientes ficheros de configuración:

- En el fichero *ldapcherry.ini* se definirá la ubicación del servidor LDAP y las credenciales del árbol base de usuarios, que en este caso corresponderán a *ou=Users, dc=vpu, dc=eps, dc=uam, dc=es*.

- En el fichero *attributes.yml* se establecen los campos necesarios para que el formulario cree correctamente el usuario.

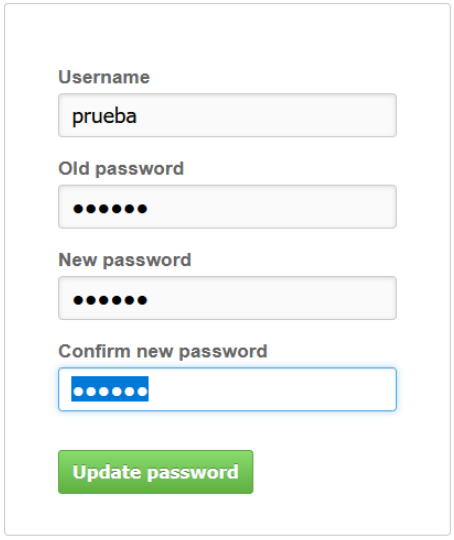


Figura 4.8: Formulario de cambio de contraseña.

- La configuración en *roles.yml* permite asignar grupos específicos a nuevos usuarios en el momento de la creación. Por ejemplo, para otorgarles permisos de administración.

Para iniciar *LDAPCherry* al arrancar en contenedor se creará una unidad de systemd:

```
# cat /etc/systemd/system/ldapcherry.service
[Unit]
Description=LDAPCherry Startup unit

[Service]
Type=simple
ExecStart=ldapcherryd -c /etc/ldapcherry/ldapcherry.ini -D

[Install]
WantedBy=multi-user.target

# systemctl daemon-reload && systemctl enable ldapcherry
  && systemctl enable ldapcherry
```

4.7. Documentación de los procedimientos

Todos los servicios implementados han sido documentados en forma de *wiki* en un contenedor basado en una plantilla de *MediaWiki* con la finalidad de que el trabajo técnico realizado pueda ser retomado por terceras personas (Figura 4.10). Asimismo, se ha elaborado otro contenedor similar cuya documentación está orientada al usuario final de los servicios. En ella se especifican los procedimientos necesarios para conectarse a una VPN o la utilización de directorios compartidos.

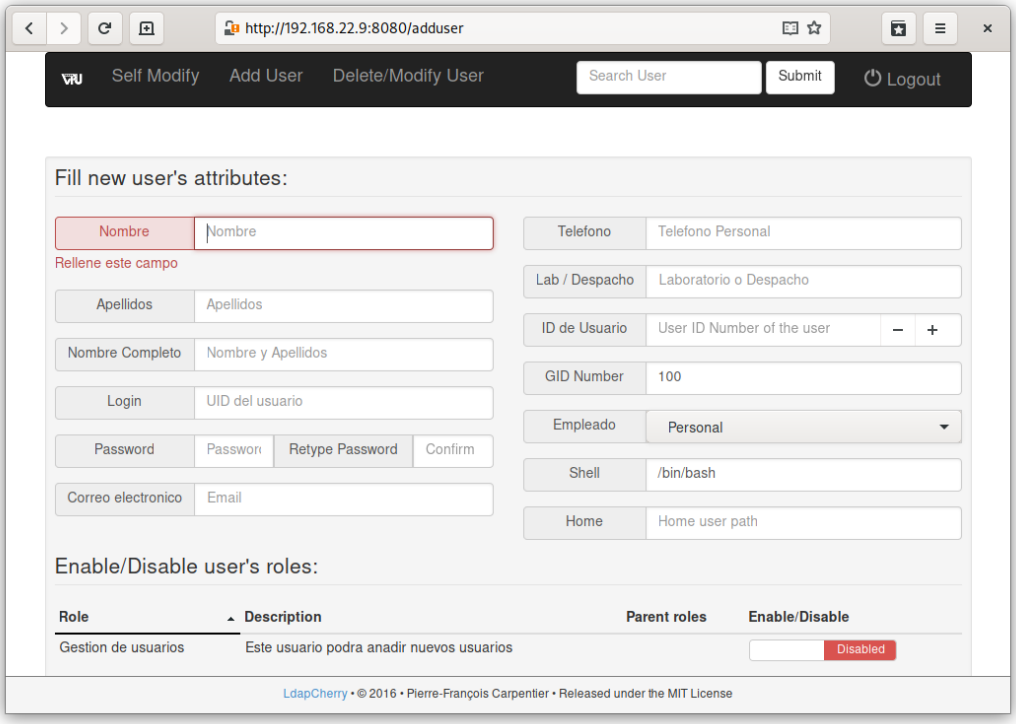


Figura 4.9: Formulario LDAPCherry para añadir usuarios.

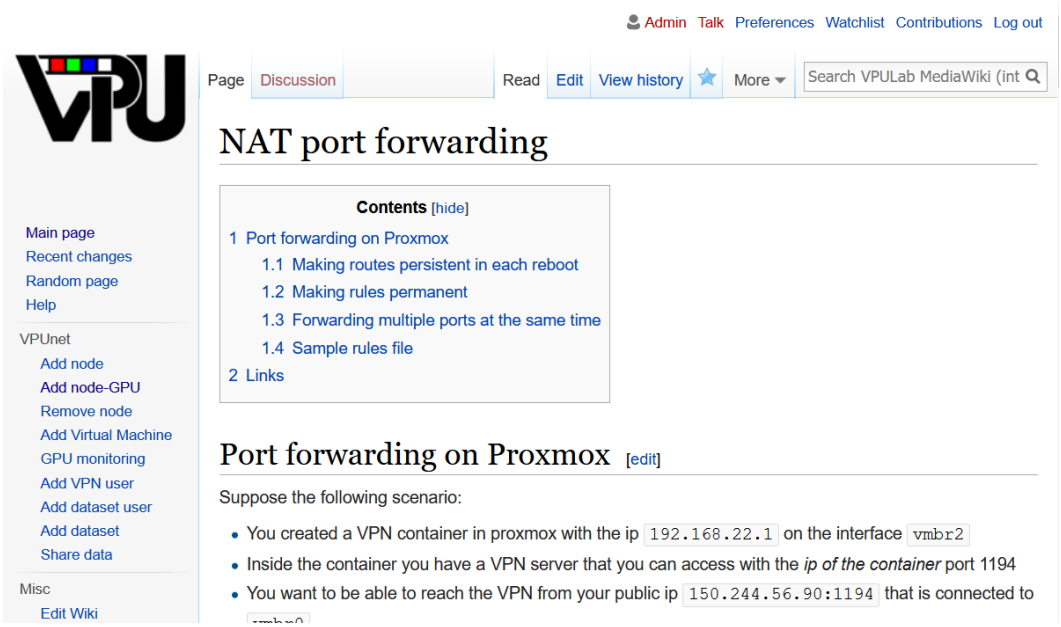


Figura 4.10: Página de la *wiki* destinada a la documentación técnica.

Capítulo 5

Evaluación

5.1. Introducción

A continuación se realizarán diversas pruebas para verificar que el sistema en su conjunto funciona correctamente y puede ser utilizado en un entorno real. Además de las pruebas meramente técnicas, también se realizarán pruebas de los principales casos de uso a los usuarios del sistema.

5.2. Pruebas y resultados

5.2.1. Hardware

5.2.1.1. Velocidad del switch

Para comprobar que la velocidad de los *switches* instalados es la adecuada, se hará uso de una herramienta llamada *iperf* con la que se realizará una operación de transferencia entre los dos primeros servidores de gestión, utilizando como servidor de *iperf* la máquina situada en 192.168.21.2 y ejecutando el siguiente comando en la máquina situada en 192.168.21.3:

```
# iperf -c 192.168.21.2 -i 1 -n 10G -yC > switch_throughput.csv
```

En la Figura 5.1 se observa que las velocidades cumplen con lo esperado en el *switch* utilizado, velocidades que son cercanas a 1Gbps y que aprovechan las características del hardware.

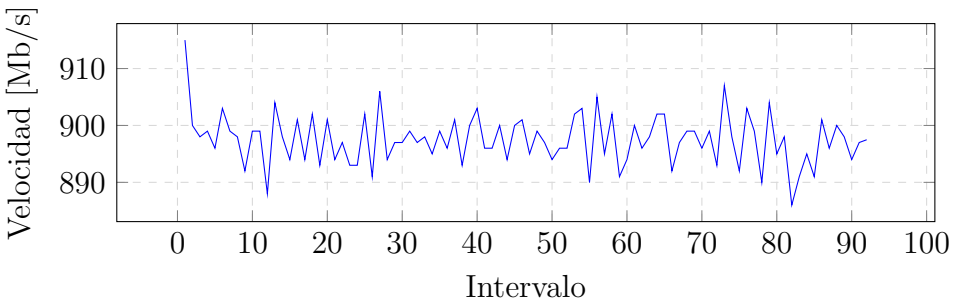


Figura 5.1: Velocidad de transferencia de un fichero de 10GB durante 90 segundos desde 192.168.21.2 hacia 192.168.21.3.

5.2.1.2. Pruebas de conectividad de los servidores

Para comprobar si todos los servidores son accesibles y verificar su tiempo de respuesta, se ha realizado un *ping* en distintas máquinas. Para cada máquina se ha recopilado su tiempo de respuesta durante cien peticiones y se han elaborado los siguientes gráficos. En la Figura 5.2 se puede visualizar en los cuatro primeros casos el comportamiento a través de las conexiones directas de los *switches* instalados. En los dos últimos casos la latencia es mayor debido al salto adicional desde la puerta de enlace hasta la interfaz virtual del contenedor.

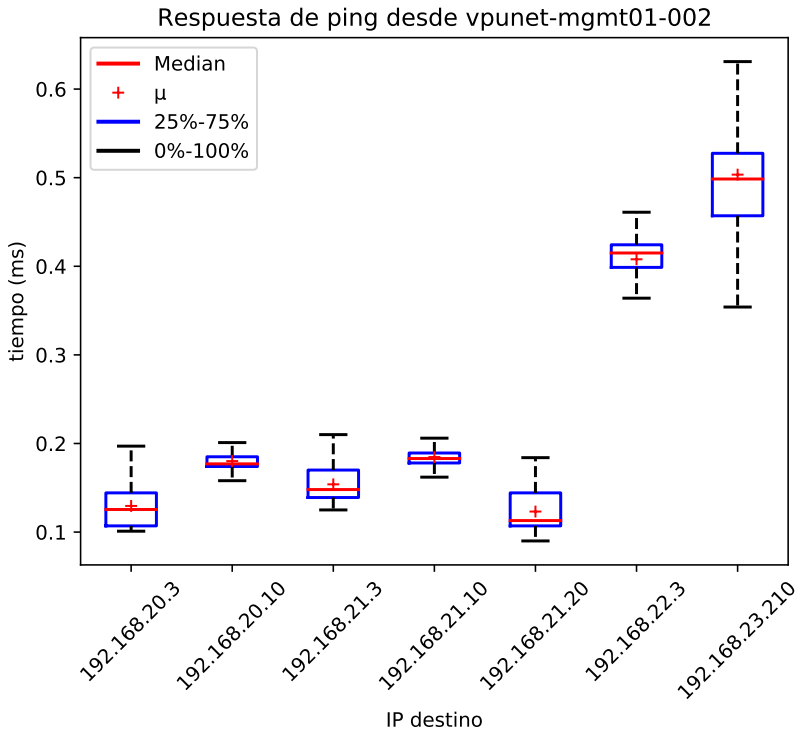


Figura 5.2: Pruebas de conectividad hacia las subredes principales desde *vpunet-mgmt01-002* (192.168.21.2 y 192.168.21.2).

A continuación, se han realizado pruebas de conectividad para comprobar que el enrutado al resto de subredes es correcto a través de la VPN. Al contrario que en las pruebas anteriores, todos estos nuevos casos mostrados en la Figura 5.3 presentan más latencia que en la figura anterior ya que todo el tráfico tiene que dirigirse a través del túnel VPN produciéndose un nuevo salto de red.

5.2.2. Pruebas de disco

5.2.2.1. Pruebas de las unidades ZFS en el servidor de discos

El servidor de discos es la máquina que figura como *vpunet-stg01-010* y que está situada en un clúster junto al resto de las máquinas de gestión como se puede ver en la Figura 3.6. En el servidor de discos existirá un conjunto de 5 discos que se montarán en */mnt/stg-pool1* y un conjunto de 9 discos que se montarán en */mnt/stg-pool2*. Para realizar pruebas de escritura, será necesario medir la escritura real del disco y no los valores de transferencia a la RAM, que es donde ZFS envía temporalmente

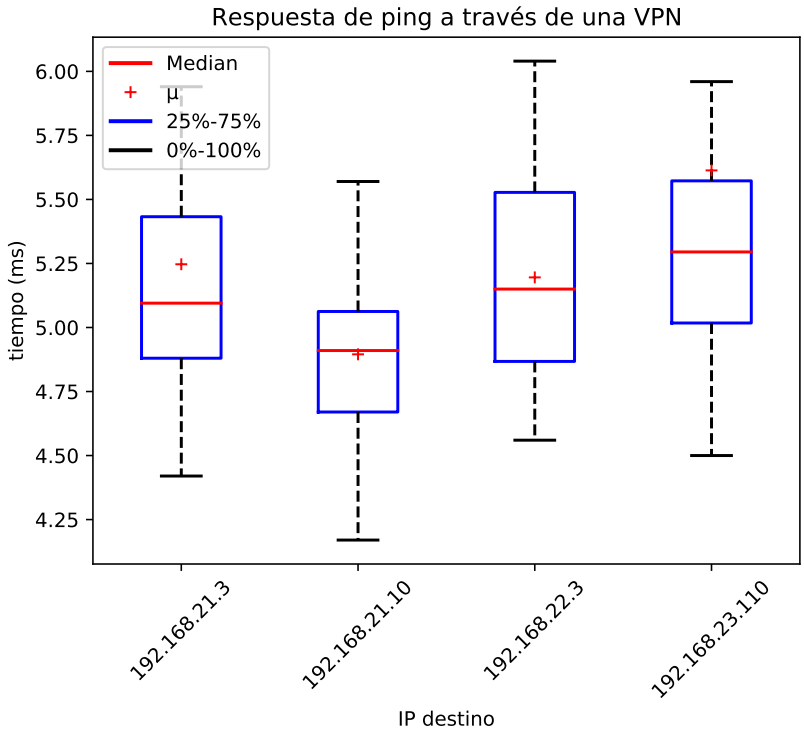


Figura 5.3: Pruebas de conectividad con la VPN configurada hacia todas las subredes disponibles.

la información antes de escribirla. Para ello se ha hecho uso del software *bonnie++*, que permite realizar este tipo de medidas. En consecuencia, será necesario realizar las pruebas con un fichero del doble de la RAM del sistema, que es de 32GB. Además, el comando con el que se realizarán las pruebas será el siguiente:

```
# bonnie++ -s 64296 -n 0 -f -b -u root
```

En la Tabla 5.1 se pueden visualizar los resultados para el pool *stg-pool1* con distintos parámetros de ZFS. Los parámetros *ashift* 12 y 13 varían el tamaño del bloque del disco a 2KB y 4KB respectivamente mientras que los parámetros *raidz1* y *raidz2* varían la redundancia en uno y dos discos respectivamente.

Tasas de transferencia para stg-pool1			
		Escritura (MB/s)	**Lectura (MB/s)**
raidz1	**ashift=12**	**267**	**515.5**
	ashift=13	265.5	509
raidz2	**ashift=12**	259	462.5
	ashift=13	259	443.5

Tabla 5.1: Velocidades de transferencia realizadas mediante *bonnie++* con distintas configuraciones de alineación de bloques (ashift 12 y 13) y distintas configuraciones de redundancia (raidz1 para redundancia 1 y raidz2 para redundancia 2).

Seguidamente, se tomó la configuración que mejores resultados otorgaba (ashift=12 con raidz1) y se habilitó el algoritmo de compresión LZ4, realizando la misma prueba con *bonnie++*. Los resultados de la Tabla 5.2 muestran como la compresión mejora

Comparativa de compresión de stg-pool1			
	Escritura (MB/s)	**Lectura (MB/s)**	**% CPU**
Sin compresión	267	515.5	**8 %**
Compresión LZ4	**548**	**1200**	10 %

Tabla 5.2: Mediciones de velocidades de transferencia y consumos de CPU realizadas mediante *bonnie++* con y sin compresión.

notablemente las tasas de transferencia. De esta manera, siempre que se disponga de RAM y procesador suficiente, como en el de esta máquina dedicada, es preferible tener habilitada la opción de compresión LZ4 en el sistema de archivos.

Finalmente, con los cambios realizados en *stg-pool1*, se ofrece en la Tabla 5.3 una comparativa a modo de resumen de los dos *pools* del sistema. Como línea de mejora se propone realizar las pruebas y optimizaciones para aumentar el rendimiento también en *stg-pool2*.

Velocidades en los pools de vpunet-stg01-010		
	Escritura secuencial	**Lectura secuencial**
stg-pool1	548,5 MB/s	1200 MB/s
stg-pool2	364,5 MB/s	306,5 MB/s

Tabla 5.3: Velocidades de transferencia realizadas con *bonnie++* en el equipo *vpunet-stg01-010* para los *pools* de disco presentes en el mismo. Ambos pools poseen en esta tabla la misma configuración, con los parámetros ashift=12 (2KB/bloque, por defecto) y raidz2 (redundancia de 2 discos).

5.2.2.2. Pruebas de transferencia a través de máquinas virtuales

En las opciones de almacenamiento de Proxmox, se ofrecen distintos tipos de imágenes de disco para el almacenamiento en máquinas virtuales. Se han realizado pruebas de transferencia con todos los tipos de caché disponibles con el formato *raw*. Para ello, se ha creado una máquina virtual basada en Ubuntu con dos unidades de disco: una de ellas orientada al sistema operativo y la segunda orientada a las pruebas de disco. La unidad de pruebas, consiste en una sola partición de 32 gigabytes formateada como ext4. Posteriormente se monta dicha unidad en */mnt* y se realizan las pruebas pertinentes con la utilidad *bonnie++* instalada en la máquina virtual. En este caso, se recomienda realizar las pruebas con un fichero con el doble de la memoria RAM, para evitar problemas con el cacheado de los datos, por esa razón se ha utilizado un tamaño de fichero de 1 gigabyte, el doble de la memoria instalada, 512 megabytes.

```
# bonnie++ -d /mnt -s 1G -n 0 -f -b -u root
```

Además de medir la velocidad del disco virtual con diferentes tipos de caché, se analiza la velocidad del disco real en el que están alojadas las imágenes virtuales para poder comparar las diferencias de rendimiento en virtualización y en nativo. Estos resultados se recogen en la Figura 5.4.

El modo de cacheado por defecto en las máquinas virtuales de Proxmox es el modo *sin caché*. Estas pruebas confirman la evidencia de que el modo *sin caché* es el más apropiado para las máquinas virtuales, ya que es el que menos compromete la velocidad

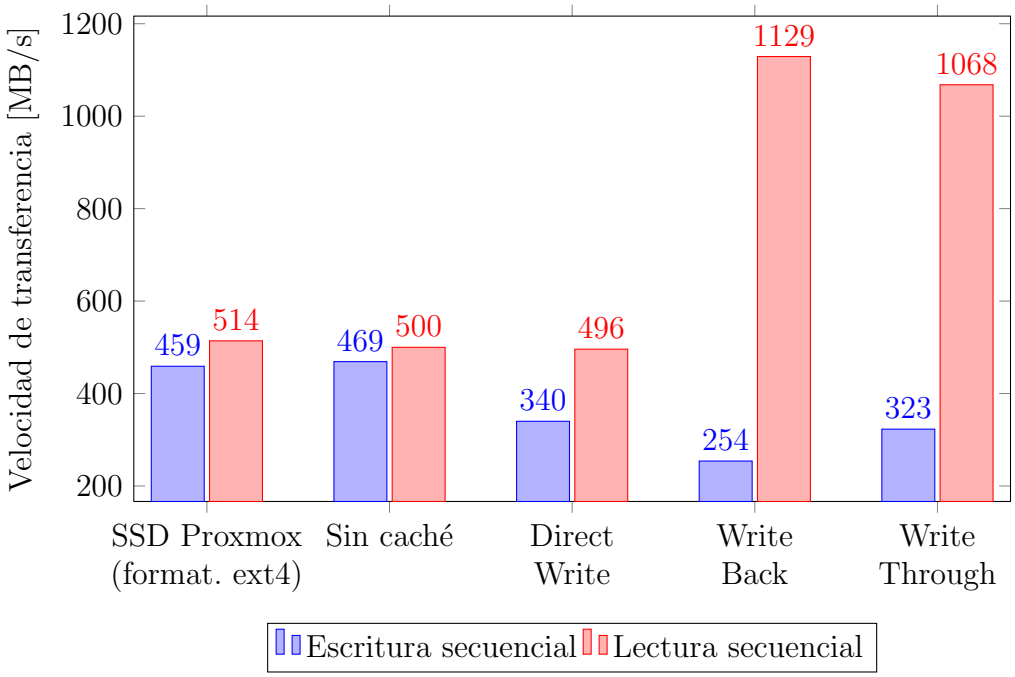


Figura 5.4: Pruebas de escritura secuencial para una misma máquina virtual con distintos tipos de caché.

de escritura. En los modos *Write Back* y *Write Through* se observa el efecto de la caché en las tasas de lectura.

5.2.2.3. Pruebas de transferencia de copias de respaldo

El escenario de estas pruebas consiste en realizar un respaldo de una máquina virtual con un tamaño de disco determinado. Existirán dos pruebas de este tipo: En la primera, el respaldo se transferirá por red al servidor de discos donde finalmente se almacenará. En la segunda, el respaldo se realizará de forma local. Será necesario crear una imagen de disco de prueba para llenarla de datos aleatorios con *dd*, ya que, de lo contrario, sólo se transferirían los datos correspondientes al uso de disco en la imagen virtual y no el tamaño completo de la misma, debido a las características propias de la imagen de disco. En la Tabla 5.4 se presentan los resultados de los experimentos realizados con 10, 50 y 300GB respectivamente.

	qcow2		**raw**	
	NFS	**Local**	**NFS**	**Local**
10GB	3m 27s	1m 59s	2m 25s	1m 42s
50GB	10m 2s	5m 29s	11m 19s	5m 44s
300GB	1h 5m 43s	34m 27s	1h 8m 39s	31m 16s

Tabla 5.4: Tiempos de copia y compresión del respaldo de una imagen virtual en los formatos *raw* y *qcow2* según tamaño de disco.

Extrapolando estos casos a un escenario de 800GB a respaldar por máquina y 10 máquinas, el tiempo de un respaldo completo ascendería aproximadamente a un total de 30h y 15m. Estos tiempos podrían reducirse implementando otras técnicas como concurrencia o respaldos incrementales.

5.2.3. Casos de prueba de los servicios y herramientas utilizados

Para asegurar que la usabilidad del software es buena y que los procedimientos para la realización de tareas son correctos, se han propuesto los siguientes cinco casos de prueba que cubren los escenarios principales de uso del sistema:

- Creación de un usuario por parte de un administrador.
- Acceso a la VPN del laboratorio desde una red externa.
- Acceso a los recursos compartidos de la red.
- Acceso remoto a los escritorios multiusuario.
- Formulario destinado a usuarios para el cambio de contraseña.

Estos casos de prueba figuran de manera más detallada en el Anexo B. Todas las pruebas han sido superadas con éxito por parte de varios integrantes del laboratorio. De esta manera, estos procedimientos se consideran preparados para un escenario real y habitual.

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

El trabajo tiene como objetivo construir una infraestructura fiable apta para el trabajo en el laboratorio de investigación. La red ha sido diseñada con el objetivo de que sus componentes sean redundantes ante fallos del sistema. Se ha tenido en cuenta en esta red su posible extensibilidad separando cada funcionalidad de forma lógica, haciendo posible que la red pueda mantenerse y ampliarse en un futuro sin ser reemplazada. De manera paralela a este trabajo se ha elaborado una documentación en forma de *wiki* para que cualquier elemento de la infraestructura pueda ser reemplazado o mejorado afectando lo menos posible en otras funcionalidades de la red, gracias a la arquitectura modular de ésta.

Aunque se han empleado técnicas de clusterización, no se ha llegado a exprimir completamente su potencial, debido al escaso número de nodos. Una posible mejora de este aspecto es la habilitación de un clúster de alta disponibilidad que, ante el fallo de un nodo, permita la recuperación de las máquinas virtuales en ejecución en un nodo adyacente, minimizando de esta manera la posibilidad de un escenario de pérdida de conectividad de una máquina.

A la vista de los resultados, se puede afirmar con seguridad que se puede llevar a cabo una arquitectura de red de alto rendimiento utilizando exclusivamente tecnologías libres. Todos y cada uno de los aspectos de la red han sido establecidos gracias a proyectos de código abierto, proyectos que ya tienen una larga trayectoria y reputación, que permiten establecer una arquitectura completa a coste cero. Además de los aspectos puramente técnicos, se ha hecho especial hincapié en que la utilización de los servicios por parte de los usuarios finales sea lo más sencilla posible, para ello se han simplificado las interfaces de gestión de usuarios y se han elaborado guías de utilización que quedan a disposición de todos los integrantes del laboratorio.

6.2. Trabajo futuro

La infraestructura construida es un punto de partida sólido en el que es posible realizar ampliaciones que se han contemplado pero que están fuera del alcance de este trabajo. Conforme a los resultados obtenidos, se han tenido en cuenta las siguientes mejoras a la estructura de la red del laboratorio:

- En un futuro a corto plazo se realizará la instalación de nodos de trabajo a los que se les añadirán GPUs para realizar tareas de procesamiento. A estas máquinas se les realizará una instalación limpia de Proxmox y se estudiará la posibilidad de utilizar contenedores en lugar de máquinas virtuales para poder suministrar el recurso compartido de la GPU entre varios usuarios de forma simultánea.

- Para comprobar que el estado de las GPUs es el correcto, se contemplará la integración de los sistemas de detección de temperatura de las tarjetas gráficas con la herramienta de monitorización que se implementó en el Capítulo 5.

- Se realizarán pruebas más intensivas de las configuraciones de los *pools* del servidor de discos con distintos parámetros de ZFS para buscar el mejor rendimiento, se evaluarán las velocidades de cada disco individual para detectar anomalías.

- Como ya se ha mencionado anteriormente, al final de este trabajo se ha trabajado con técnicas de clusterización para facilitar la transferencia y el respaldo de máquinas virtuales entre nodos de la forma más rápida posible. Una mejora de esta técnica es la inclusión de la alta disponibilidad en el clúster, que ante la falla de un nodo permita su recuperación en los nodos restantes y minimizar el tiempo en el que un servicio se encuentra fuera de línea.

- En la Sección 5.2.2.2 se realizaron múltiples pruebas de transferencia de disco a través de una máquina virtual con 512MB de memoria asignados. Sería necesario realizar las mismas pruebas con un incremento en el tamaño de la memoria para verificar si hay alguna diferencia de rendimiento con estas variaciones.

- Por último, sería interesante explorar la funcionalidad de respaldos incrementales, esto es, que en lugar de realizar copias íntegras de una máquina, sólo se respalda aquello que ha cambiado respecto a la última copia realizada. Gracias a esto, se consigue una reducción significativa del espacio utilizado en disco y una menor saturación de la red.

Bibliografía

[1] P. Li, "Centralized and decentralized lab approaches based on different virtualization models," *J. Comput. Sci. Coll.*, vol. 26, p. 263–269, Dec. 2010. 4

[2] E. Kaloshina, "Proxmox ha virtualization cluster," 2017. 4

[3] W. Ahmed, *Mastering Proxmox: Build virtualized environments using the Proxmox VE hypervisor*. Packt Publishing, 2017. 6

[4] R. Hertzog and R. Mas, "The debian administrator's handbook," *Freexian SARL*, 2013. 16

[5] K. Saini, *Squid Proxy Server 3.1: Beginner's Guide*. Packt Pub., 2011. 17

[6] B. Marques and M. Ricardo, "Synchronization of application-driven wsn,," *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, p. 3, 02 2017. 18

[7] M. Butcher, *Mastering OpenLDAP: Configuring, Securing, and Integrating Directory Services*. From technologies to solutions, Packt Pub., 2007. 19

[8] J. Keijser, *OpenVPN Cookbook*. Packt Publishing, 2017. 20

[9] W. Barth, *Nagios, 2nd Edition: System and Network Monitoring*. No Starch Press, 2008. 21

[10] D. Wojsław, *Introducing ZFS on Linux: Understand the Basics of Storage with ZFS*. Apress, 2017. 23

Glosario de Términos

- **VPS**: *Virtual Private Server*
- **VPN**: *Virtual Private Network*
- **LDAP**: *Lightweight Directory Access Protocol*
- **NTP**: *Network Time Protocol*
- **DNS**: *Domain Name System*
- **LTS**: *Long Term Support*
- **KVM**: *Kernel-based Virtual Machine*
- **LXC**: *Linux Container*
- **NFS**: *Network File System*
- **XMPP**: *Extensible Messaging and Presence Protocol*
- **RDP**: *Remote Desktop Protocol*
- **VNC**: *Virtual Network Computing*
- **SSH**: *Secure SHell*
- **SSD**: *Solid State Drive*
- **HDD**: *Hard Disk Drive*
- **SAS**: *Serial Attached SCSI Drive*
- **RAM**: *Random Access Memory*
- **LZ4**: *Lempel Ziv 4*
- **ZFS**: *Zettabyte File System*
- **EXT4**: *Extended File System 4*
- **RAID**: *Redundant Array of Independent Disks*
- **GPU**: *Graphical Processing Unit*

Apéndices

Apéndice A

Plataformas de virtualización

A.1. VMware ESXi

ESXi (*https://www.vmware.com/es/products/esxi-and-esx.html*) es un hipervisor bare-metal que se instala como único sistema residente en la máquina. ESXi funciona con un núcleo Linux basado en la distribución RHEL de Red Hat. Cuenta con cliente web para administrar las máquinas. Tiene soporte para instantáneas y migraciones en vivo. Puede realizar copias de seguridad periódicas y tiene soporte para el sistema de archivos NFS. Cuenta también con una interfaz de administración web (Figura A.1).

El precio de la licencia para la versión *Standard* es de 967,50 € mientras que para la versión *Enterprise Plus* el precio asciende a 3.495 €. Las licencias son por cada zócalo de CPU.

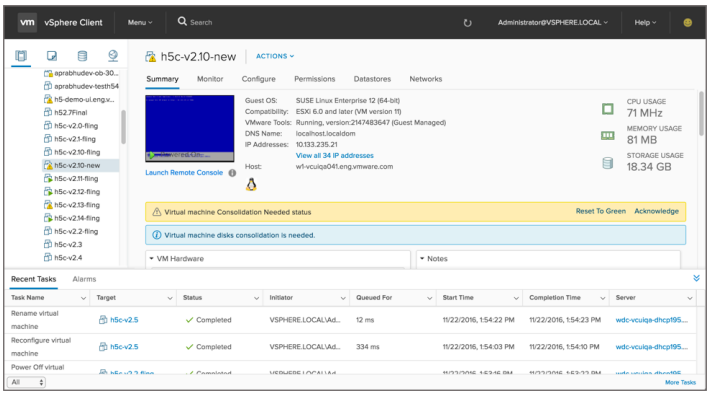


Figura A.1: *Dashboard* de administración de VMWare vSphere Client. (Fuente: *https://revistacloud.com/wp-content/uploads/2017/10/vpshere-web-client-html5.png*)

A.2. Microsoft Hyper-V

Hyper-V (*https://docs.microsoft.com/es-es/virtualization/hyper-v-on-windows/*) es el hipervisor nativo de Microsoft (Figura A.2) que se ejecuta en los sistemas Windows. Está disponible tanto en las versiones Pro de Windows como en las versiones Server. Las distintas máquinas virtuales están aisladas a modo de particiones y las llamadas a los recursos de hardware se redireccionan a través de dispositivos virtuales. Se puede administrar tanto de forma gráfica como por línea de comandos haciendo uso de *PowerShell*. También cuenta con una interfaz de administración vía web.

La mayor desventaja respecto a otros sistemas es la falta de soporte a NFS. Su mayor ventaja es su alta integración con sistemas Windows aunque también se pueden virtualizar sistemas GNU/Linux dentro de él.

El precio para la licencia de Windows Server 2019 es de 972$ mientras que para Windows 10 Pro el precio es de 199$.

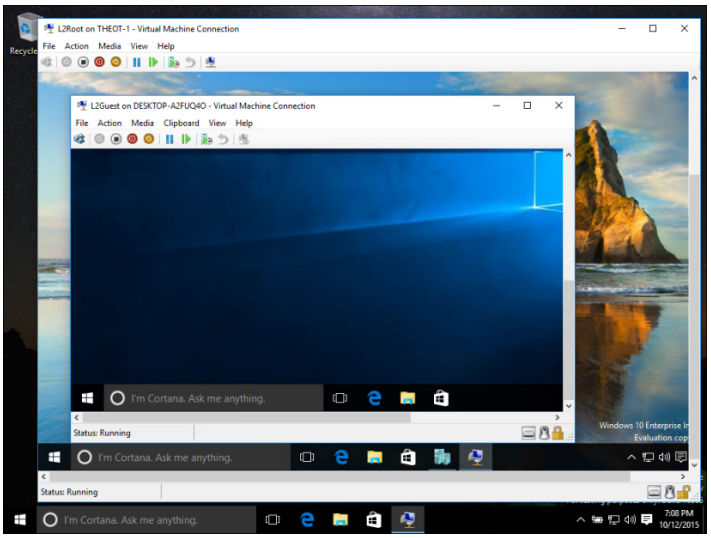


Figura A.2: Ejecución de varias máquinas virtuales con *Microsoft Hyper-V* bajo el sistema operativo *Windows 10*. (Fuente: *https://docs.microsoft.com/*)

A.3. Citrix

La suite Citrix Hypervisor incluye XenServer (*https://www.citrix.com/es-es/products/citrix-hypervisor/*), una plataforma de virtualización completa orientada a servidores (Figura A.3). *Citrix* utiliza *Xen* como hipervisor para máquinas virtuales. Se puede instalar en un entorno distribuido. *XenServer* utiliza un núcleo *Linux* modificado desde la distribución *CentOS*.

XenServer tiene versiones gratuitas y de pago, cuyas características diferenciales son el suministro de soporte y de mantenimiento.

A.4. Vmmanager

Vmmanager (*https://www.ispsystem.com/software/vmmanager*) es un sistema de la compañía ISPSystems que permite automatizar el despliegue de máquinas virtuales de KVM. Está orientado principalmente a proveedores de VPS y propietarios de *datacenters*.

Desde la administración de *Vmmanager* se pueden gestionar las máquinas virtuales (Figura A.4) así como generar imágenes de disco de las máquinas virtuales configuradas. Tiene soporte para cústers, permitiendo agrupar varios nodos en clústers lógicos.

Vmmanager puede utilizar plantillas de sistemas operativos predefinidos y es posible crear diferentes usuarios y restringir sus privilegios.

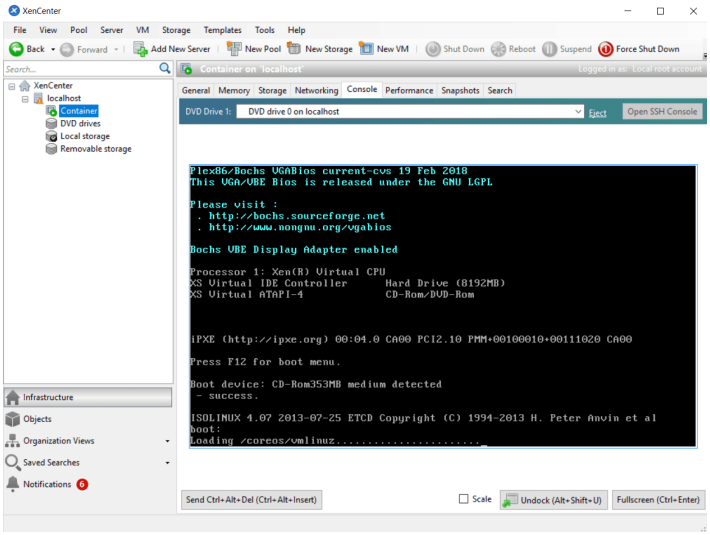


Figura A.3: Interfaz de gestión de los distintos recursos de un contenedor con *Citrix XenCenter*. (Fuente: *https://www.maquinasvirtuales.eu/*)

La arquitectura de *Vmmanager* está basada en tres contenedores de Docker: Uno de ellos se encarga de los servicios de panel de control y de los servicios de autenticación, otro para la monitorización y el último para la gestión de MySQL.

Vmmanager ofrece precios para varias máquinas, siendo el precio de la licencia por PC de 12€.

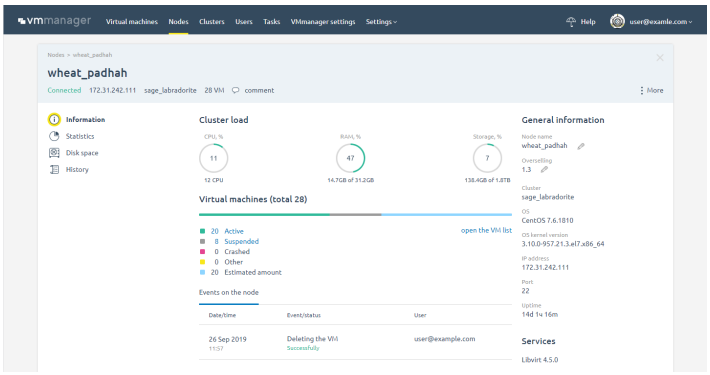


Figura A.4: *Dashboard* principal de administración de *Vmmanager*. (Fuente: *https://www.ispsystem.com/software/vmmanager*)

A.5. Virtualbox

Virtualbox (*https://www.virtualbox.org/*) es un software de virtualización desarrollado por Oracle que se ejecuta sobre un sistema operativo huésped ya existente como Windows, GNU/Linux o macOS. Por ello, no ofrece una interfaz de administración ni una gestión de usuarios (Figura A.5).

Por otro lado, cuenta con soporte de directorios compartidos entre máquinas y con compatibilidad para dispositivos USB en la máquina emulada.

VirtualBox no es un hipervisor *bare-metal* y carece de muchas funcionalidades importantes para la gestión de un sistema distribuido como pueden ser la automatización

de backups, unidades NFS o migraciones en vivo.

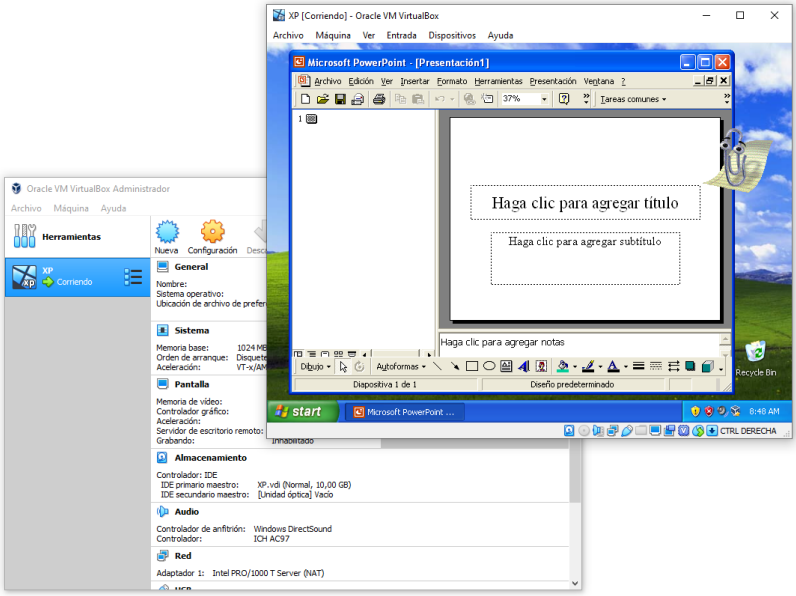


Figura A.5: Ejecución de una máquina virtual con Virtualbox instalado en un sistema Windows 10. (Fuente: elaboración propia)

A.6. KVM

KVM (*https://www.linux-kvm.org/page/Main_Page*) consiste en un módulo para virtualización dentro del kernel de Linux que permite que dicho núcleo funcione como hipervisor (Figura A.6). Requiere de un procesador con extensiones para virtualización como *Intel VT* o *AMD-V*. Ha sido portado a variedad de arquitecturas. También dispone de paravirtualización a través de *VirtIO*, que agiliza la interacción con la máquina virtual gracias al uso de dispositivos paravirtuales como puede ser la tarjeta de red o el controlador VGA usando los drivers de SPICE. Muchas plataformas de virtualización utilizan KVM para llevar a cabo la ejecución de las máquinas virtuales.

KVM fue creado a partir de una base del código de QEMU, otra plataforma de virtualización enfocada a la emulación de hardware. Al contrario que QEMU, KVM permite aceleración por hardware cuando la arquitectura es la misma en el sistema huésped que en el invitado.

A.7. Ovirt

Ovirt (*https://www.ovirt.org/*) es una plataforma libre de virtualización fundada por *Red Hat* y utilizada como base para la herramienta *Red Hat Virtualization*. Permite gestionar máquinas virtuales y recursos de manera centralizada. La interfaz de administración está basada en *Java* (Figura A.7). Soporta unidades NFS y permite la gestión de red mediante la definición de múltiples VLANs que pueden ser usadas en modo puente a través de las interfaces de red disponibles en los nodos, todo ello accesible mediante la administración web.

Ovirt puede ser ejecutado tanto en un sólo servidor o bien en un clúster de nodos.

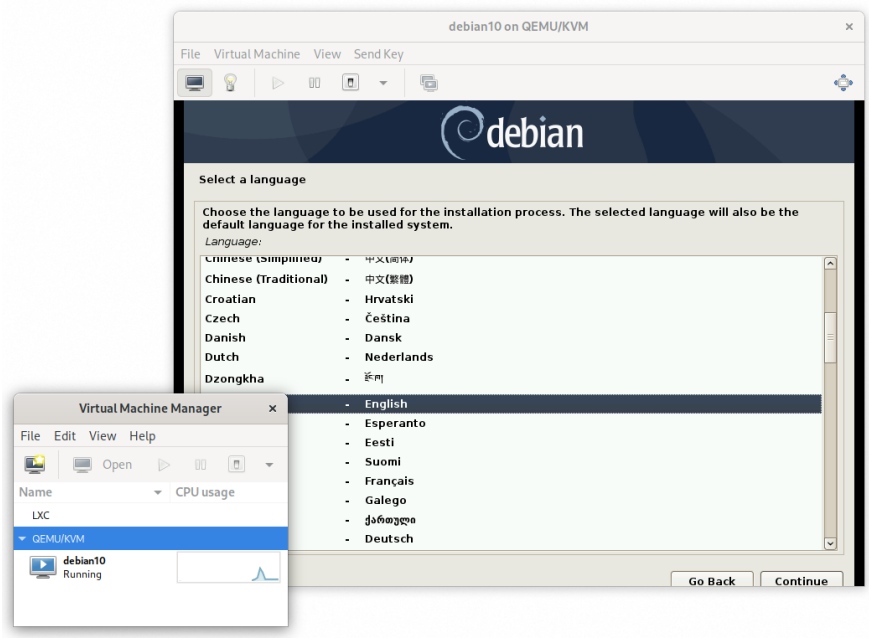


Figura A.6: *KVM* a través del *frontend* de *virt-manager*. Fuente: propia

El acceso a las máquinas virtuales está disponible mediante VNC, SPICE o RDP y se ofrece soporte para migraciones en vivo, instantáneas, clonado, plantillas e incluso recuperación ante desastres.

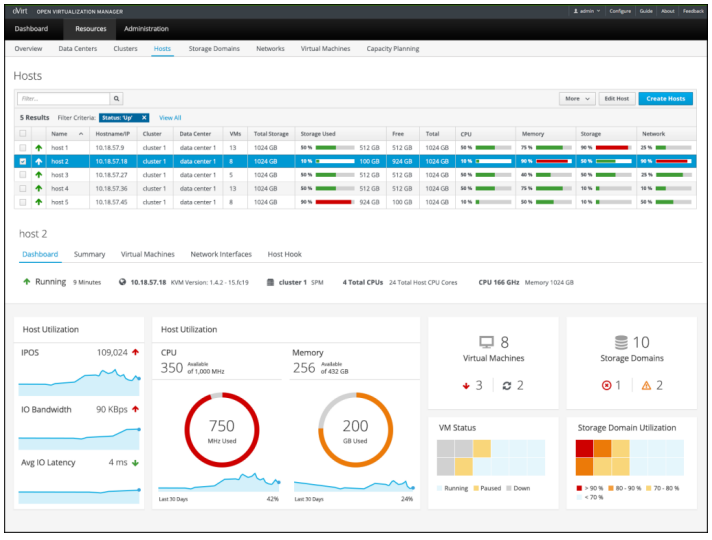


Figura A.7: Resumen de los distintos *hosts* configurados en Ovirt. (Fuente: *https://blog.ichasco.com/*)

A.8. Archipel

Archipel (*https://github.com/ArchipelProject/Archipel*) es una interfaz para la supervisión y administración de máquinas virtuales (Figura A.8). Es capaz de trabajar con los hipervisores que soporta tecnologías como LXC para contenedores o KVM, Xen y VirtualBox para máquinas virtuales.

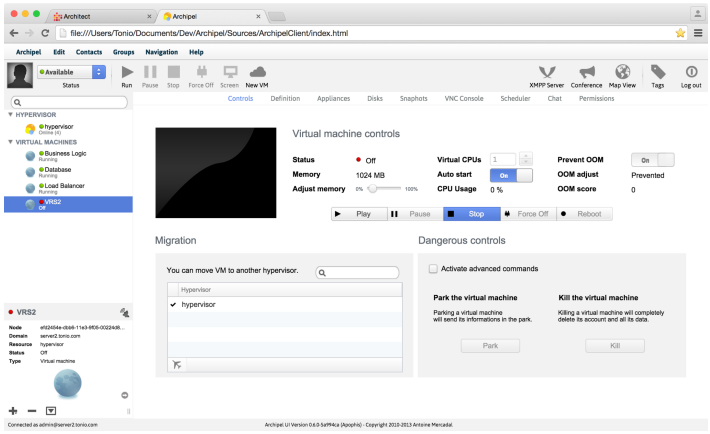


Figura A.8: Interfaz de administración principal de Archipel. (Fuente: *https://twitter.com/ArchipelProject/*)

A.9. Proxmox

Proxmox (*https://github.com/ArchipelProject/Archipel*) es una solución de código abierto de un entorno de virtualización para servidores. Está basado en la distribución Debian. El núcleo es una modificación de la rama LTS del kernel de Linux que se encuentra en las versiones de soporte a largo plazo de Ubuntu.

Proxmox utiliza virtualización a nivel de núcleo con KVM. También soporta la creación de contenedores con LXC y la gestión de clústers mediante la herramienta *corosync*.

La administración web (Figura A.9) cuenta con capacidad de gestión de usuarios y diferentes formas de representar y administrar los distintos hosts y servicios. Adicionalmente desde la interfaz se pueden realizar migraciones y backups.

Proxmox cuenta con soporte para migraciones en vivo, unidades NFS y backups periódicos.

Todas las tareas disponibles desde la administración web se pueden realizar a través de línea de comandos.

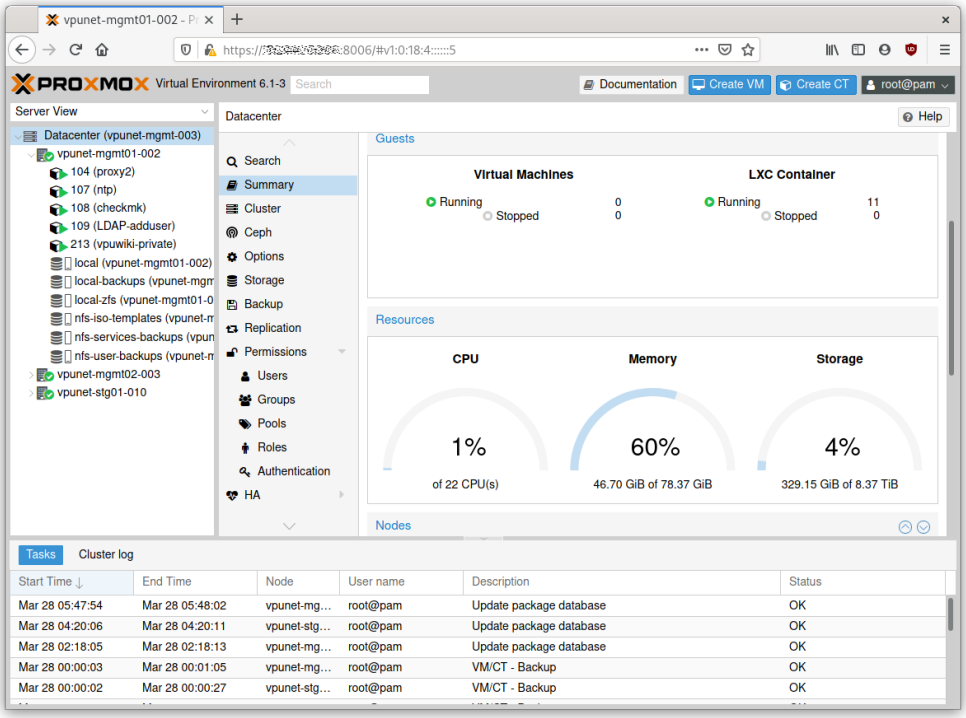


Figura A.9: *Dashboard* de administración de un clúster en Proxmox. (Fuente: elaboración propia)

Apéndice B

Casos de prueba del sistema

Para comprobar que los usuarios del laboratorio pueden utilizar correctamente el sistema se establecerá una batería de casos de prueba que verifiquen que el funcionamiento de la infraestructura es como se espera.

B.1. Creación de un usuario por parte de un administrador

Un administrador podrá realizar la creación de un usuario mediante un formulario en el que se le pedirá toda la información.

Requisitos previos:

- El usuario debe tener una cuenta creada en el servidor LDAP y formar parte del grupo *cn=man*, por lo que previamente habrá tenido que ser agregado por otro administrador.
- El usuario deberá estar conectado a la VPN para poder acceder al formulario de creación de usuarios.

Procedimientos de prueba:

- El administrador abrirá un navegador y accederá al formulario simplificado de creación de usuarios situado en *http://192.168.22.9:8080*.
- El administrador rellenará los campos de nombre, apellidos, nombre completo, login ID, contraseña, correo electrónico, teléfono, laboratorio o despacho y tipo de empleado. Seguidamente confirmará la creación. En la Figura B.1 se muestra cómo aparecen los campos a rellenar.

Postcondición:
El formulario señalará que el usuario ha sido creado correctamente. Se podrá iniciar sesión por SSH a través del comando *ssh usuario@192.168.21.10*

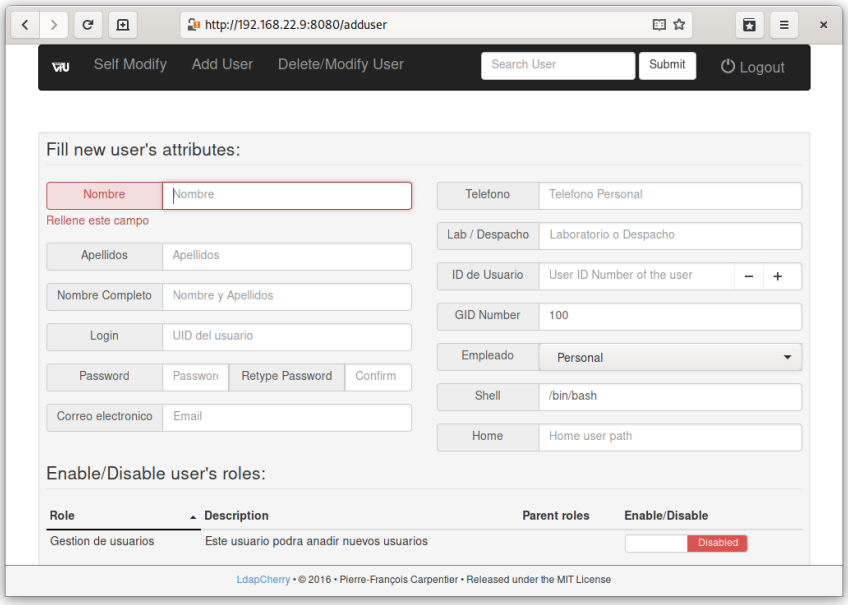


Figura B.1: Formulario de creación de usuario a través de la utilidad *CherryLDAP*.

B.2. Acceso a la VPN del laboratorio desde una red externa

Para comprobar el acceso remoto a la red por parte de los usuarios, se les pedirá que accedan a la VPN del laboratorio con sus credenciales. De esta manera, independientemente de su localización podrán utilizar la red del laboratorio.

Requisitos previos:

- El usuario debe tener una cuenta creada en el servidor LDAP.
- El usuario deberá recibir un perfil *.ovpn* con el que verificar su identidad con el servidor. Este perfil será generado y asignado por un administrador.
- El usuario deberá tener instalado el cliente de OpenVPN en el sistema operativo Windows 10.

Procedimientos de prueba:

- El usuario instalará el perfil asignado en su ordenador. Para ello seleccionará la opción *Import File* del desplegable del icono de *OpenVPN* de la bandeja del sistema (Figura B.2).
- Por último se conectará a la VPN, accediendo mediante el desplegable a la opción *Conectar* del nombre del perfil instalado. (Figura B.3).

Postcondición:
El cliente de *OpenVPN* no muestra ningún error. Es posible hacer *ping* a las IPs 192.168.24.1 y 192.168.22.3. De esta manera se asegura que además del servicio VPN, el funcionamiento del enrutado hacia otras subredes es correcto.

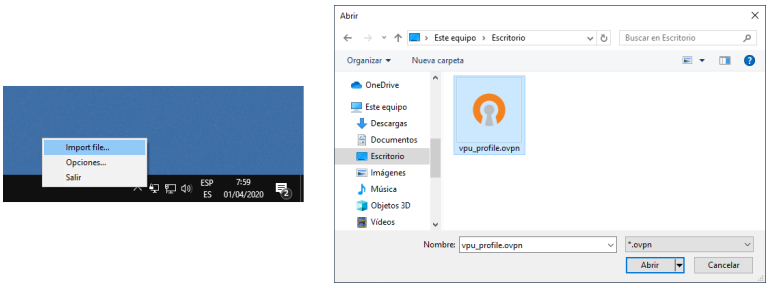


Figura B.2: Importación del perfil al cliente OpenVPN en Windows 10.

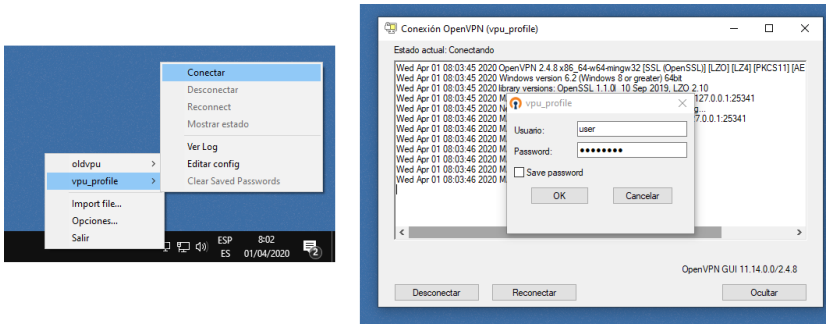


Figura B.3: Conexión a la VPN en Windows 10.

B.3. Acceso a los recursos compartidos de la red

Este proceso es necesario para que los usuarios puedan usar los recursos comunes y los directorios personales de cada usuario.

Requisitos previos:

- El usuario debe tener una cuenta creada en el servidor LDAP.
- El usuario deberá estar dentro de la UAM o utilizar la VPN de la misma.
- El usuario deberá tener instalado el cliente *Filezilla*.

Procedimientos de prueba:

- El usuario abrirá *Filezilla* e introducirá como servidor la dirección *sftp://150.244.214.237* en el puerto especificado, además de sus credenciales. (Figura B.4).
- El usuario podrá visualizar las unidades compartidas y creará o moverá un fichero en su directorio *home*.

Postcondición:

Tras crear o mover un fichero, la transferencia del mismo habrá finalizado y se mostrará en el directorio personal del usuario que ha iniciado sesión. El usuario no podrá acceder a otros directorios personales que no sean los suyos.

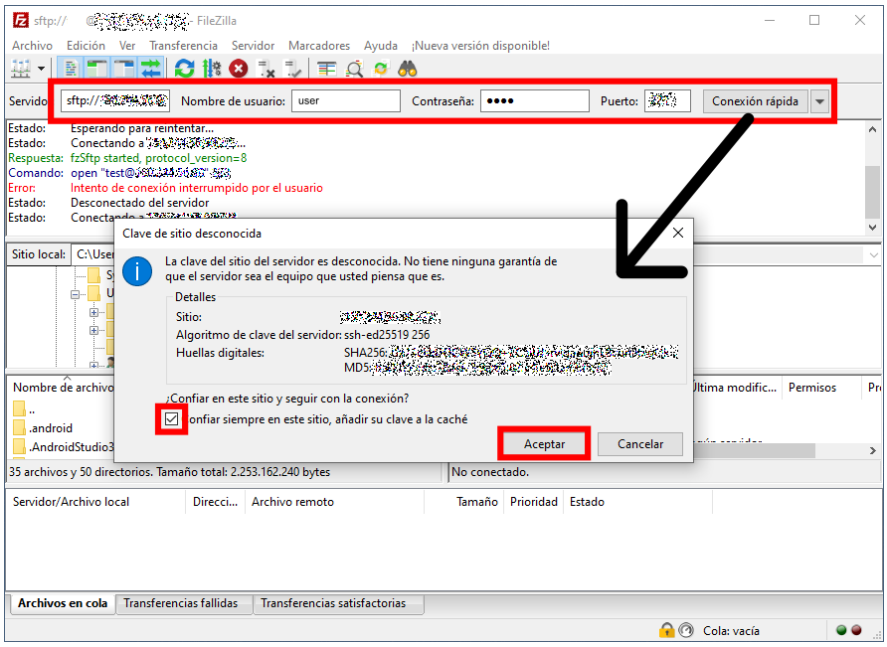


Figura B.4: Conexión a las unidades de red compartidas a través de *filezilla*.

B.4. Acceso remoto a los escritorios multiusuario

Requisitos previos:

- El usuario debe tener una cuenta creada en el servidor LDAP.
- El usuario deberá utilizar la VPN del laboratorio.
- El usuario deberá estar autorizado para utilizar la máquina, para ello, un administrador tendrá que agregarle al grupo correspondiente.
- El usuario debe utilizar Windows 10.

Procedimientos de prueba:

- El usuario abrirá desde el Menú Inicio el programa *Conexión a Escritorio remoto*.
- El usuario introducirá la dirección IP de la máquina remota, en este caso, 192.168.23.110 y se conectará, tras haber aceptado la alerta de seguridad mostrada en la Figura B.5.
- El usuario introducirá sus credenciales de LDAP para el inicio de sesión de la Figura B.6 y se mostrará el escritorio.

Postcondición:
Tras mostrarse el escritorio. El usuario podrá controlar la máquina y lanzar aplicaciones en ella. Si abre un navegador podrá visitar páginas web asegurando que el tráfico *http* es redirigido hacia el Proxy.

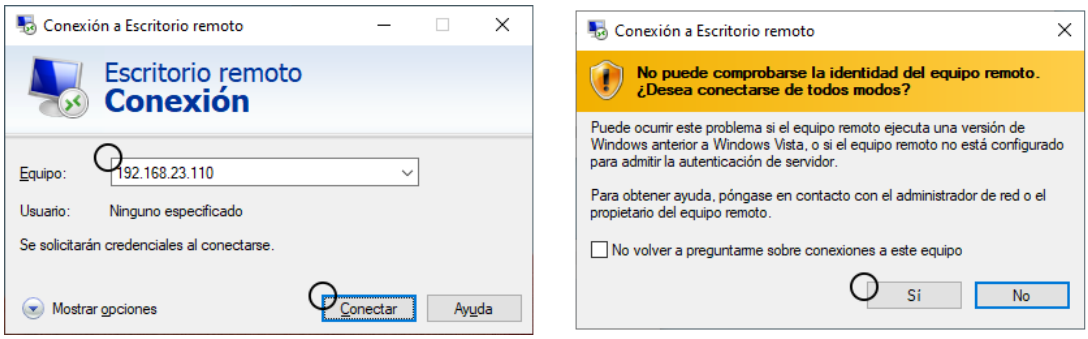


Figura B.5: Proceso de conexión al escritorio multiusuario. El usuario introduce la dirección de la máquina remota y confirma la conexión.

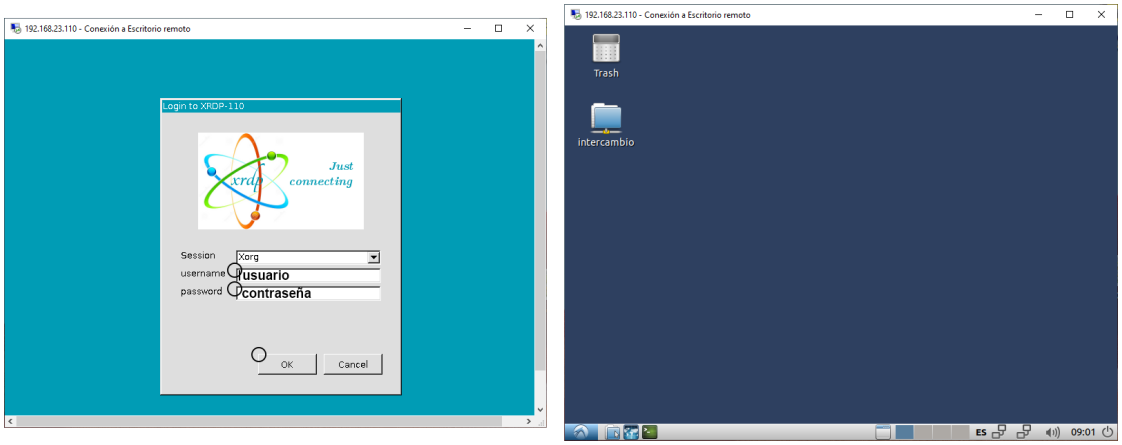


Figura B.6: Proceso de identificación al escritorio multiusuario, primero el usuario introduce sus credenciales y, tras el inicio de sesión, se lanza el escritorio del usuario.

B.5. Formulario destinado a usuarios para el cambio de contraseña

Requisitos previos:

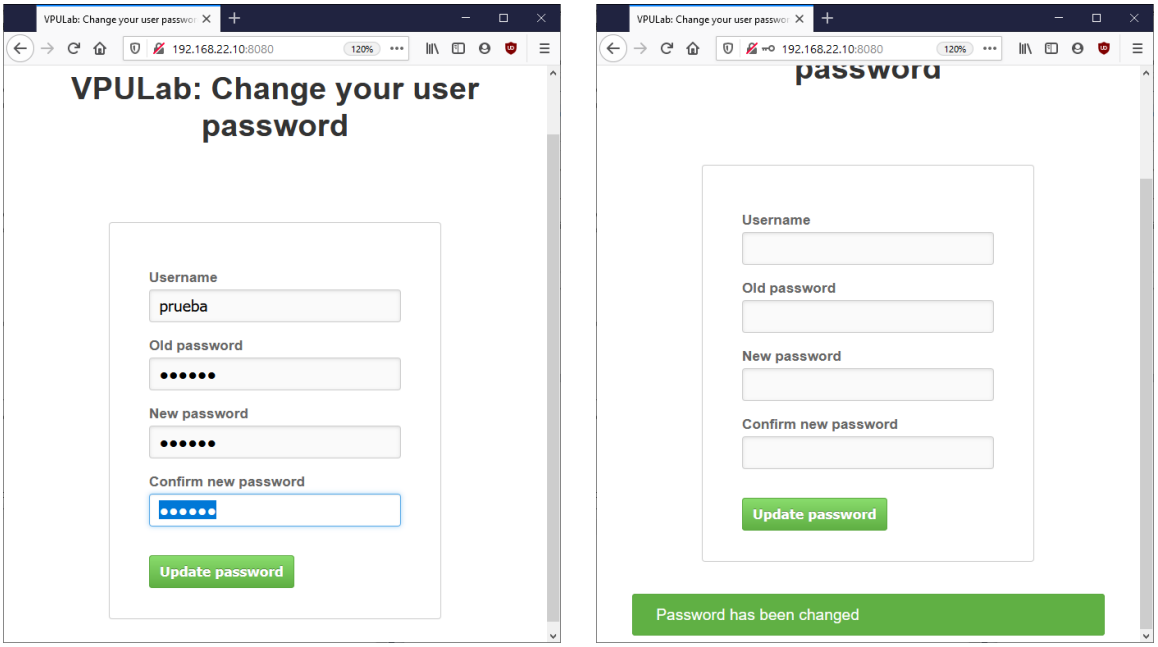
- El usuario debe tener una cuenta creada en el servidor LDAP.
- El usuario debe conocer su antigua contraseña antes de cambiarla.
- El usuario deberá utilizar la VPN del laboratorio.

Procedimientos de prueba:

- El usuario accederá con un navegador a http://192.168.22.10:8080/.
- El usuario rellenará los campos de nombre de usuario, contraseña antigua, y los dos últimos campos con la nueva contraseña, que debe tener más de ocho caracteres, tal y como se muestra en la Figura B.7a.
- El usuario recibirá una respuesta afirmativa cuando el cambio es realizado correctamente, como en el caso de la figura B.7b.

Postcondición:

Tras efectuar el cambio de contraseña, el usuario podrá identificarse con sus nuevas credenciales a través de SSH en la IP 192.168.21.10.



(a) Completado del formulario. (b) Confirmación de cambio de contraseña.

Figura B.7: Formulario de cambio de contraseña destinado a usuarios.